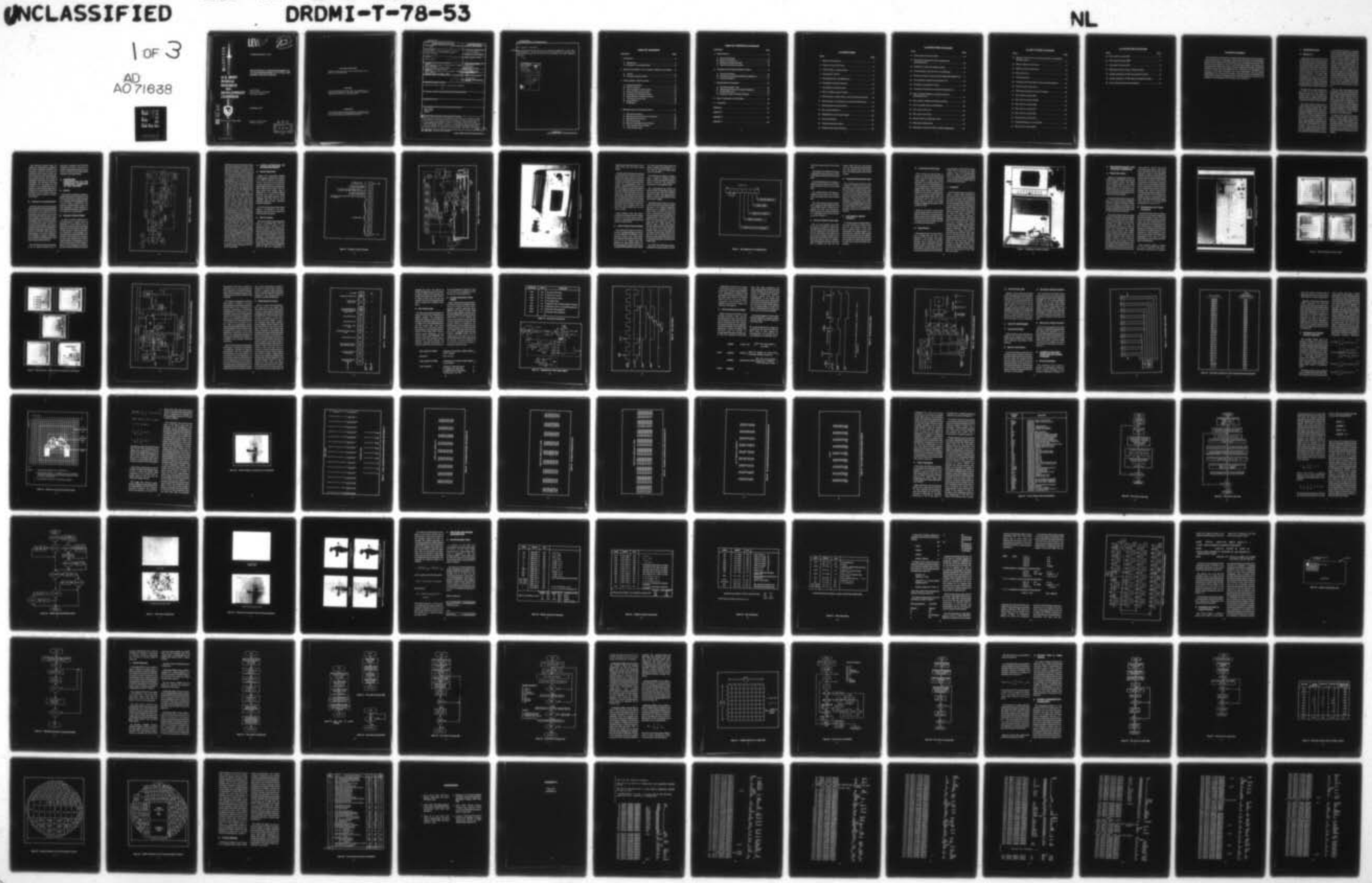
AD-A071 638 ARMY MISSILE RESEARCH AND DEVELOPMENT COMMAND REDSTO--ETC F/G 9/2
APPLICATION OF A MICROPROGRAMMED, BIT SLICE MICROPROCESSOR TO T--ETC(U)
SEP 78 L G MINOR

UNCLASSIFIED DRDMI-T-78-53 NL

1 OF 3

AD
A071638



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

LEVEL II

AD A071638

TECHNICAL REPORT T-78-53

APPLICATION OF A MICROPROGRAMMED, BIT SLICE MICROPROCESSOR TO THE REAL TIME IMAGING TRACKER PROBLEM

U.S. ARMY MISSILE RESEARCH AND DEVELOPMENT COMMAND

Lewis G. Minor
Advanced Sensors Directorate
Technology Laboratory

30 September 1978

Redstone Arsenal, Alabama 35809

Approved for Public Release;
Distribution Unlimited.

DDC FILE COPY

DDC
RECEIVED
JUL 25 1979
D

DMI FORM 1000, 1 APR 77

DISPOSITION INSTRUCTIONS

DESTROY THIS REPORT WHEN IT IS NO LONGER NEEDED. DO NOT RETURN IT TO THE ORIGINATOR.

DISCLAIMER

THE FINDINGS IN THIS REPORT ARE NOT TO BE CONSTRUED AS AN OFFICIAL DEPARTMENT OF THE ARMY POSITION UNLESS SO DESIGNATED BY OTHER AUTHORIZED DOCUMENTS.

TRADE NAMES

USE OF TRADE NAMES OR MANUFACTURERS IN THIS REPORT DOES NOT CONSTITUTE AN OFFICIAL ENDORSEMENT OR APPROVAL OF THE USE OF SUCH COMMERCIAL HARDWARE OR SOFTWARE.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER T-78-53	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Application of a Microprogrammed, Bit Slice Microprocessor to the Real Time Imaging Tracker Problem.		5. TYPE OF REPORT & PERIOD COVERED Technical Report.
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Lewis G. Minor	1T161101A91A	8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Commander US Army Missile Research and Development Command ATTN: DRDMI-TE Redstone Arsenal, Alabama 35809		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS DA-PROJECT NO. 1T1.61101A91A AMCMS-611101.91A.0011
11. CONTROLLING OFFICE NAME AND ADDRESS Commander US Army Missile Research and Development Command ATTN: DRDMI-TI Redstone Arsenal, Alabama 35809		12. REPORT DATE 30 September 1978
		13. NUMBER OF PAGES 255
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)	259p.	15. SECURITY CLASS. (of this report) UNCLASSIFIED
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release. Distribution Unlimited.

DRDMI-T-78-53

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

SOLID STATE
IMAGE DEVICE
TRACKER

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

Following sections detail the design and construction of an imaging tracker built inhouse. The program was very successful and nearly all of the objectives in the original scope of work were achieved. The program's major accomplishments are listed as follows: (1) Designed, built, and tested the first Army imaging tracker capable of correlation and adaptive gate optical contrast tracking, (2) the feasibility of trackers using high speed microprocessors and CCD TV technology was demonstrated, (3) several correlation → next page
(over)

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

393 427

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

20. Abstract (Continued)

cont.

and template matching techniques were studied and implemented in real time microprocessors. Though the tracker was built primarily as a research tool, the hardware concept has the potential for achieving a very small packaging size.

Accession For	
NTIS GRA&I	☒
DDC TAB	☐
Unannounced	☐
Justification____	
By____	
Distribution/	
Availability Codes	
Dist	Avail and/or special
A	

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

ILLUSTRATIONS

ILLUSTRATIONS (Continued)

ILLUSTRATIONS (Continued)

ILLUSTRATIONS (Continued)

ACKNOWLEDGMENT

The development of the tracker contained within this report could not have been achieved without the contributions of Allen Davis and Roy Lee of the Redstone Arsenal Support Agency and Richard Sims of the Technology Laboratory. Allen Davis was responsible for much of the detailed logic design in the tracker and frequently offered significant improvements to designs by the author. Allen Davis and Roy Lee had total responsibility for the construction and debugging of all hardware. Their high motivation and truly outstanding work are largely responsible for the success of the tracker. It should also be acknowledged that Richard Sims wrote and debugged most of the utility software in Section 5, and assisted in the debugging of the emulation code in Section 7.

1. INTRODUCTION

A. Background

High speed microprocessors such as the Intel 3000 bit slice microprocessor (or AMD 2901) offer the highest speeds, the most flexibility, and the greatest computing power available (excluding perhaps the Motorola 10800 ECL microprocessor) in current microprocessor technology. The high speed microprocessors are directly TTL compatible and as such can interface to TTL LSI parts. The processors can achieve from 6-10 million operations per second and are certain to have wide ranging effects on military hardware.

Though slower speed microprocessors such as the Intel 8080 and Motorola 6800 have been applied in limited ways in Army seekers, little has been done to exploit the capability of high speed bipolar microprocessors.

This report details an application of a high speed microprocessor (Intel 3000) to imaging seekers. The work clearly demonstrates that software as opposed to hardware can be used as the basis for a real time imaging tracker. In particular, two dimensional correlation and template matching normally implemented in dedicated hardware can be accomplished in real time software. The bulk of this report contains a general hardware description and detailed description of correlator software. Software for an optical contrast tracker appears in a separate report. Both trackers operate with the same hardware and the only difference between the two trackers is software changes. Thus, the software approach yields a tracker that is capable of radical changes in tracking mode. The changes can be effected merely by branching to different software. The implication is that trackers can be built that modify their tracking algorithms in response to countermeasures or battlefield conditions. Such changes could even occur after missile launch up to the time of missile and target impact. This capability has the effect of increased countermeasure immunity.

The bit slice, microprogrammed architecture of the Intel 3000 and other similar microprocessors appears to be well suited to many military applications. This follows because of the following reasons: 1) Architecture may be tailored to fit the application. 2) Microword structure can be configured to control complex hardware in parallel with microprocessor or other peripheral hardware functions. 3) The device operates at very high speeds.

In light of this report the development of future trackers will depend heavily on high speed software and hardware control. This follows since high speed software can replace much hardware. Thus, very complex tracking algorithms not practical in the past because of size and complexity are now feasible.

The following sections detail the design of a microprocessor applied to the image processing problem. The hardware structure and software techniques have general utility and can be used with any microprogrammed microprocessor. The tracking algorithms developed concentrate on correlation and template matching techniques, but the hardware with minimal or no modification can be applied to other tracking algorithms. Though the tracking algorithms were developed for the visible spectrum, they have direct application to other wavelengths.

B. Summary of Accomplishments

Following sections detail the design and construction of an imaging tracker built inhouse. The program was very successful and nearly all of the objectives in the original scope of work were achieved. The program's major accomplishments are listed as follows: 1) Designed, built, and tested the first Army imaging tracker capable of correlation and adaptive gate optical contrast tracking. 2) The feasibility of trackers using high speed microprocessors and CCD TV technology was demonstrated. 3) Several correlation and template matching techniques were studied and implemented in real time microprocessor software.

The work documented in this report is by no means complete and is expected to continue into FY78. The emphasis, however, will shift to trackers using classical pattern recognition concepts involving feature extraction and classification.

2. HARDWARE DESCRIPTION OF THE COMPUTER INTERFACE TO THE TRACKER

A. General

This section is designed to present a general description of hardware used in the interface of the tracker computer to an existing lab minicomputer. The level of detail is sufficient for a complete understanding of the software presented in later sections. A more detailed description will be given in a report now in preparation.

B. Computer Interface Details

In order to develop software for a microprocessor a means must be generated to load software into its memories. By far the most convenient method is to load memory from cards or disks using an external computer. The hardware in this system was interfaced to an existing EAI 640 minicomputer. The interface is an important part of the system since it allows tracker software development to occur with minimum turn around. Figure 1 shows a block diagram of the microprocessor memory interface to the EAI

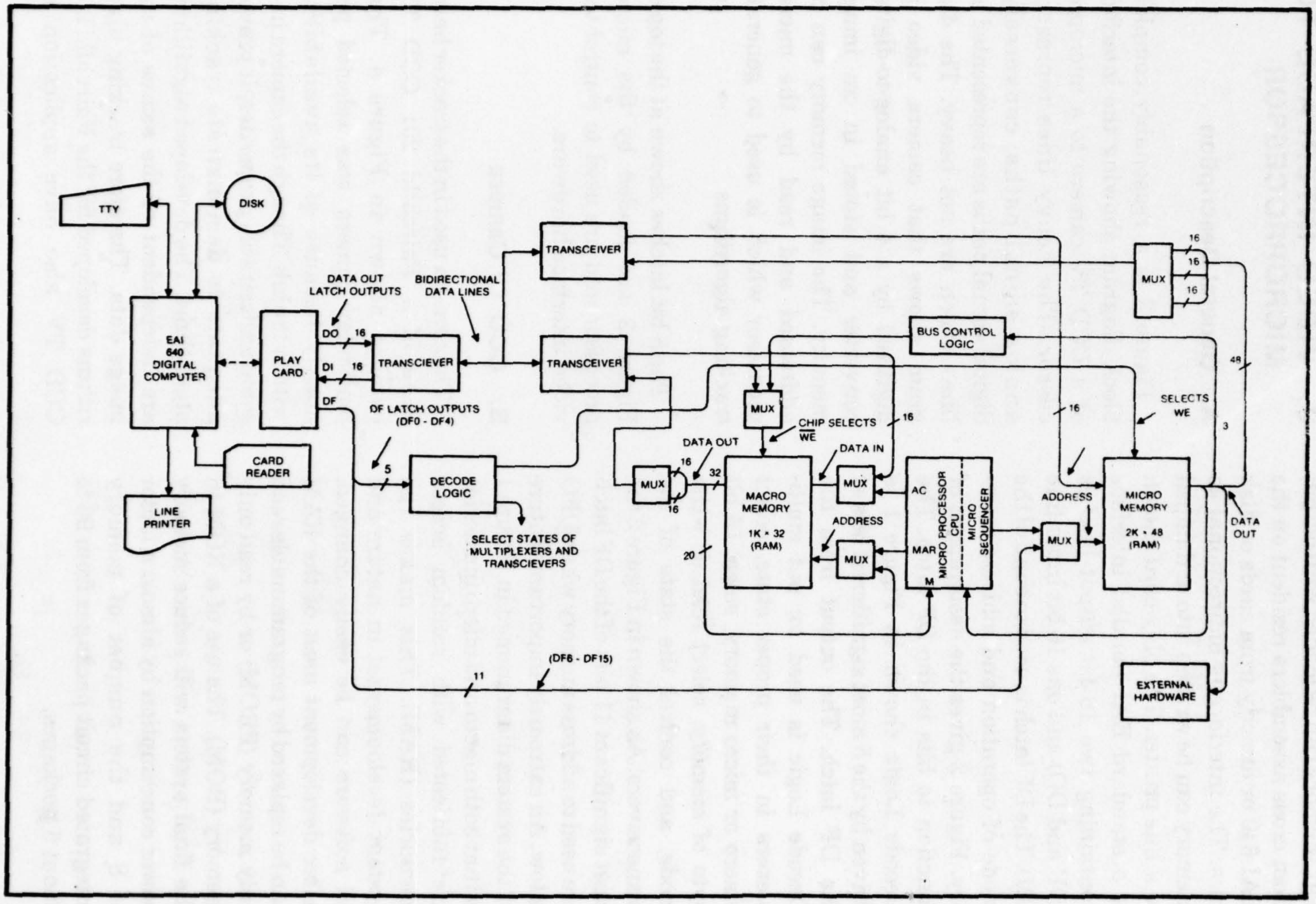


Figure 1. Memory block diagram.

640. Memory can be loaded with output from cross assemblers resident on the EAI 640 or directly from cards or disk files. The interface is bidirectional so memory can be written into or dumped to a line printer. The play card shown is a standard EAI parallel interface containing two 16 bit output latches (DF and DO) and one 16 bit input line (DI). The DF latch is used to control the mode of operation and address memory. Figure 2 gives the assignment of function to bits in the DF latch. The Decode Logic shown in Figure 1 is driven by the 5 most significant bits of the DF latch. The output from the Decode Logic is used to: put multiplexers in their proper state, select macro or micro memory, select 16 bit byte of memory, select read or write mode, and control the state of the transceivers. As shown in Figure 2 the least significant 11 bits of the DF latch are used to address memory when DFO is low. An extremely important feature of the system diagrammed in Figure 1 is that both macro and micro memories are fabricated with random access memories (RAM). This makes the system developmental in nature and all software can be easily changed. After development most of the RAM can be replaced by programmable read only memory (PROM) or by read only memory (ROM). The use of a ROM in the final system will reduce memory power consumption by almost a factor of 8 and the number of memory integrated circuit packages from 96 to about 6 packages.

3. VIDEO INTERFACE TO MICROPROCESSOR

A. General Description

Figure 3 is a reasonably complete block diagram showing the interface of a CCD TV camera to a microprocessor. The heavy lines represent analog signal paths; conversely, digital signal paths are represented by lines which are not heavy. The diagram shows that camera video is digitized by a 4 bit analog-to-digital converter and stored in an image memory. The image memory can be addressed and read by the microprocessor which is used to generate tracking algorithms.

The 8 bit latches shown at the top of Figure 3 are loaded by the microprocessor and are used to control the video interface hardware.

B. CCD TV Camera

The camera used in the tracker hardware is a Fairchild 201 CCD TV camera shown in Figure 4. This particular camera was selected primarily because of its availability within the lab. Though the camera has some undesirable properties, it proved adequate to demonstrate tracking algorithms. The developed algorithms are independent of the source of the image data. Therefore tracking algorithms developed for the Fairchild 201 CCD TV also have application to

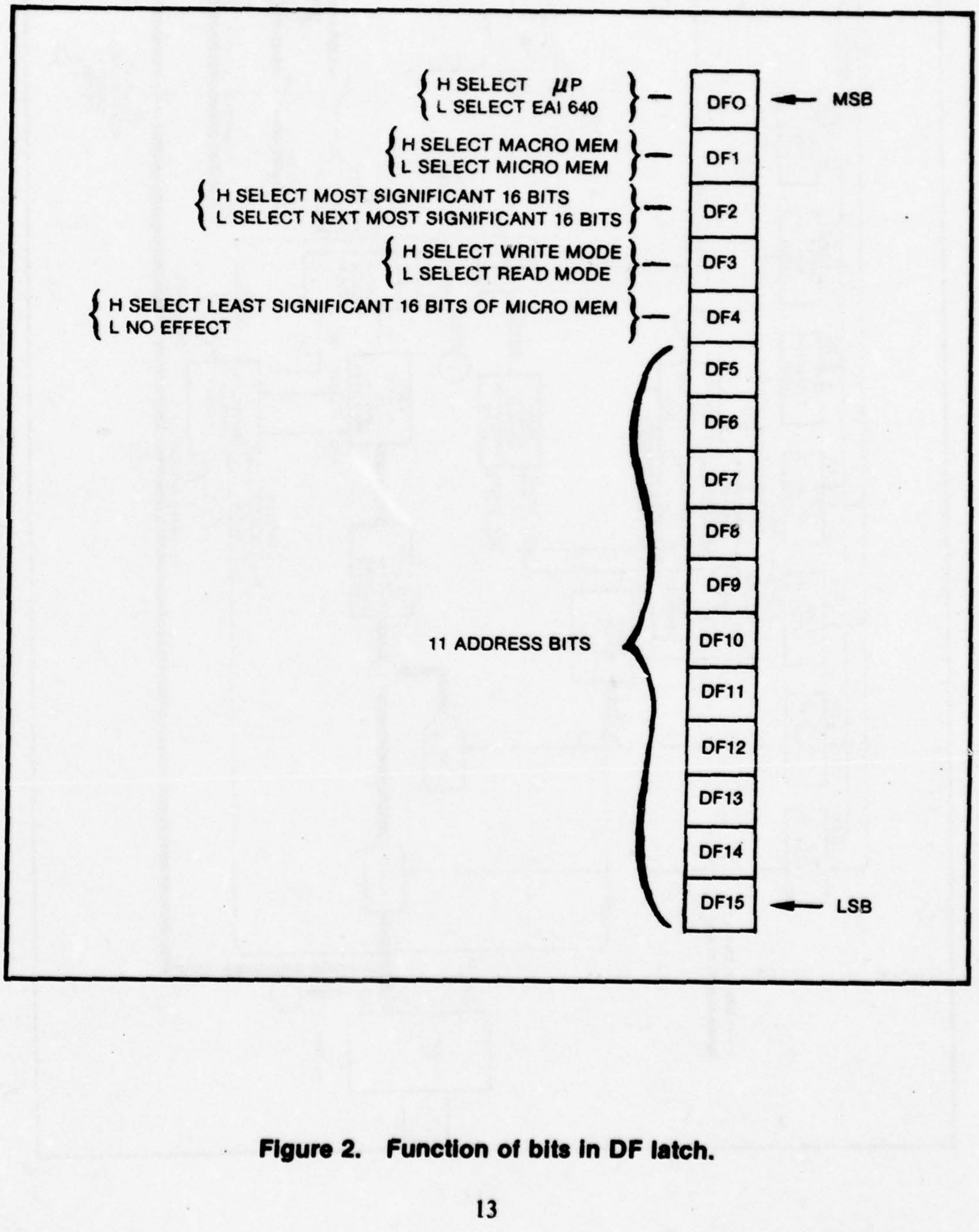


Figure 2. Function of bits in DF latch.

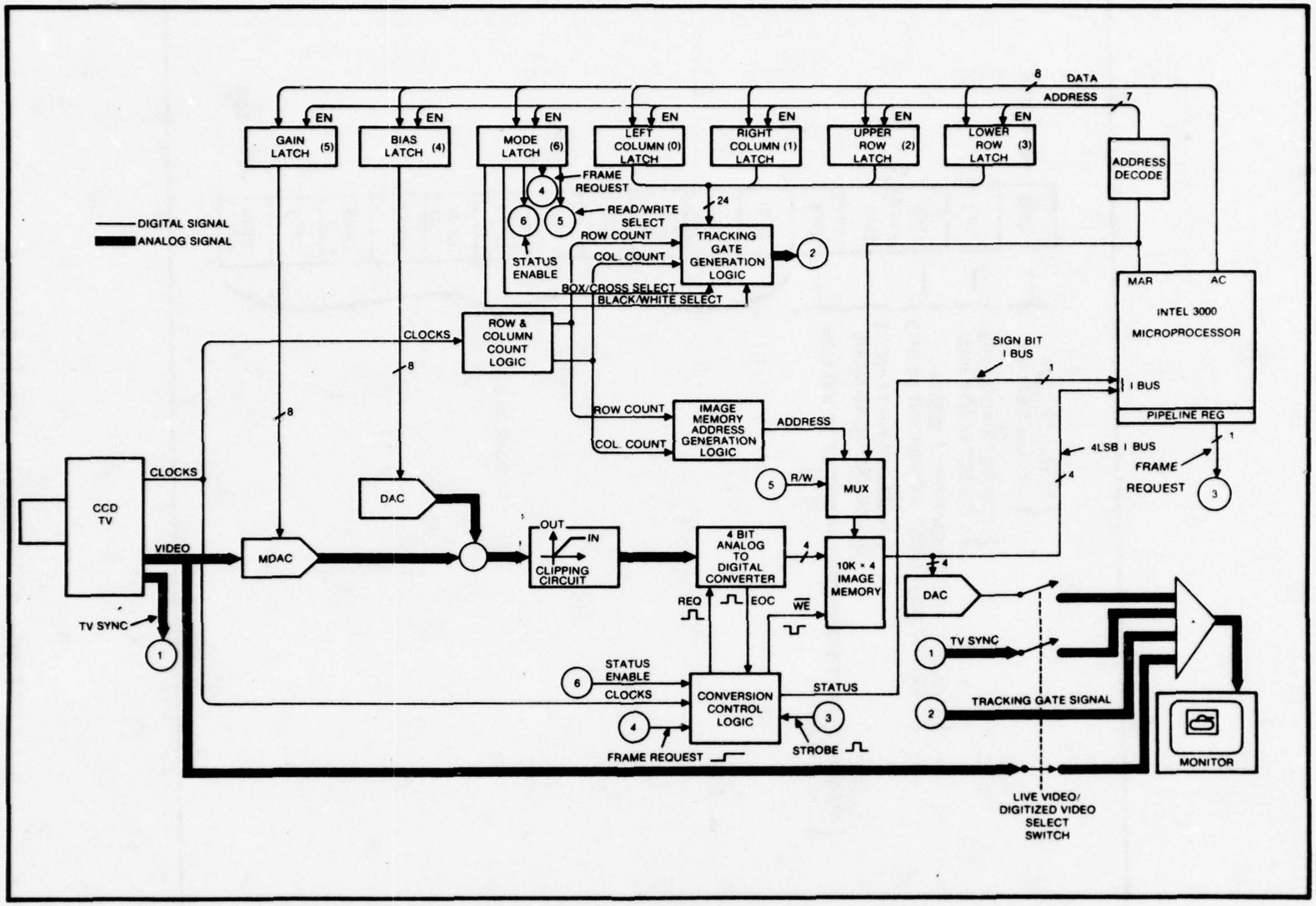


Figure 3. Video interface to microprocessor.

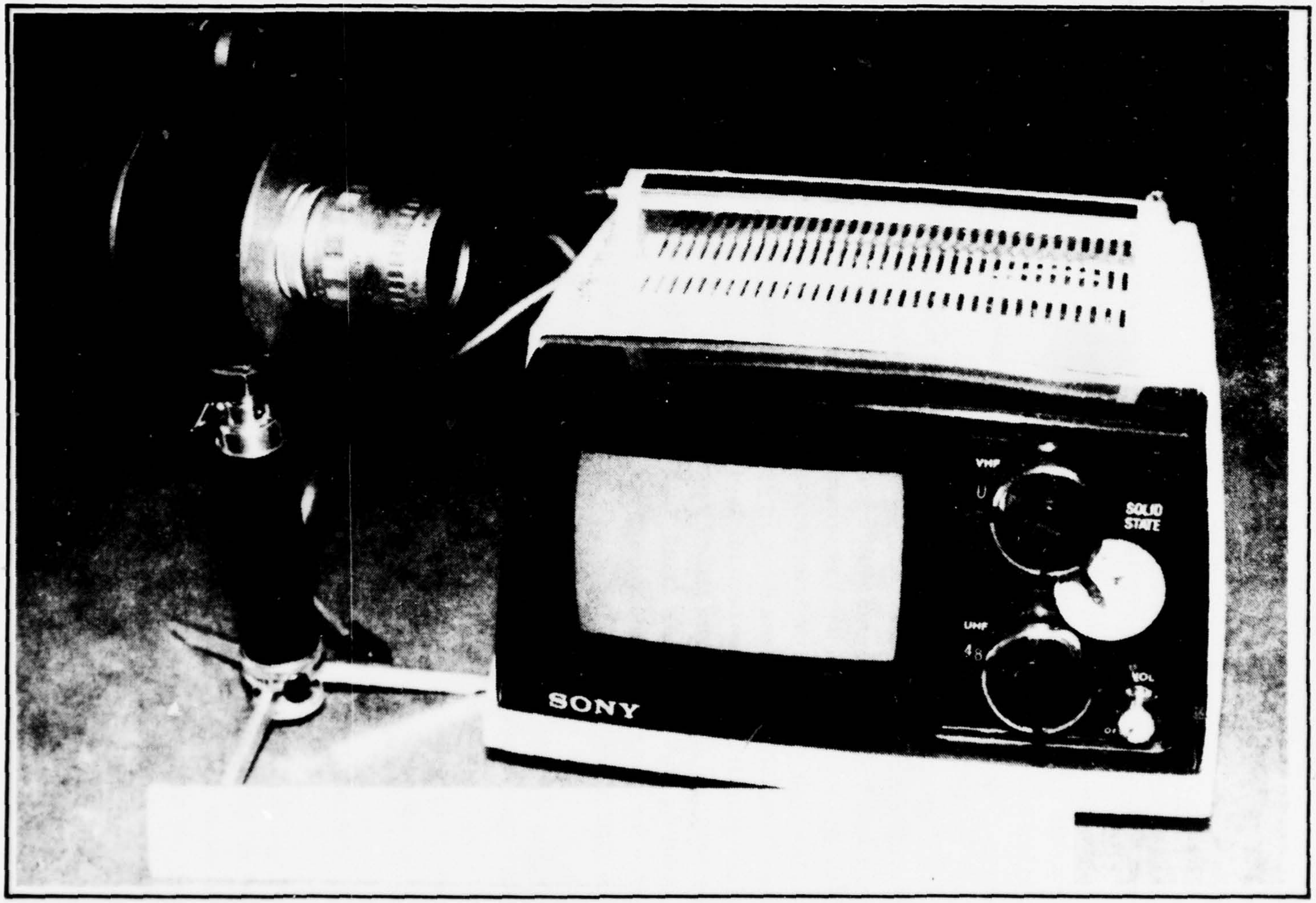


Figure 4. Fairchild 201 CCD TV.

imaging cameras operating in the 3-5 micro meter and 8-14 micro meter bands.

The Fairchild camera operates in the visible spectrum at a frame rate of 123 frames/sec with a resolution 100 x 100 elements. The camera uses 2:1 interlace and has non-standard composite video output. The non-standard video required a modified TV monitor to view the TV image, and made recording of the video difficult with standard video recorders. Video recordings could have been made with specialized recorders but the high cost of the units forced the use of available recorders. Recordings of marginal quality have been made of test results with an Ampex recorder by adjusting the recording head rate servo for a fairly stable lock on every 4th sync pulse.

Some problems were also experienced from: the tendency of the camera to bloom on bright objects, nonuniformity in response (pixel to pixel), and camera produced AGC.

C. Video Interface Control Latches

The GAIN latch, Figure 3, is used to control a multiplying digital to analog converter (MDAC). The MDAC in turn controls the gain of the video amplifier. The gain is selectable in the range from 0.0 to 5.0. Higher gains are possible but usable gains depend on the responsivity variation from pixel to pixel and fixed pattern noise. For the CCD 201 solid state image sensor the photo response nonuniformity is on the order of +15% and higher gains were not indicated.

The BIAS latch, Figure 3, is used to drive a digital to analog converter (DAC). In this case the DAC output is used to "buck out " DC components appearing in the amplified video.

By manipulating the GAIN and BIAS latches it is possible to optimize the video going to the 4 bit analog to digital converter. The clipping circuit shown is used to protect the A/D from over range voltages.

The MODE latch, Figure 3, contains 5 bits of data which are used to control: the state of the address multiplexer (MUX shown in Figure 3), the gate color (black or white), the gate type (box or crosshairs), enable status, and frame requests. Figure 5 shows the bit assignments in the MODE latch. When the microprocessor is reading data from the image memory, the MUX is used to select the memory address register of the microprocessor as the source for image memory address. When status is enabled, the conversion control logic will force the sign bit of the I bus high after a new frame has been stored since the strobe in the pipeline register is enabled in the control logic.

The LEFT COLUMN latch, Figure 3, defines the column location (1 to 100)

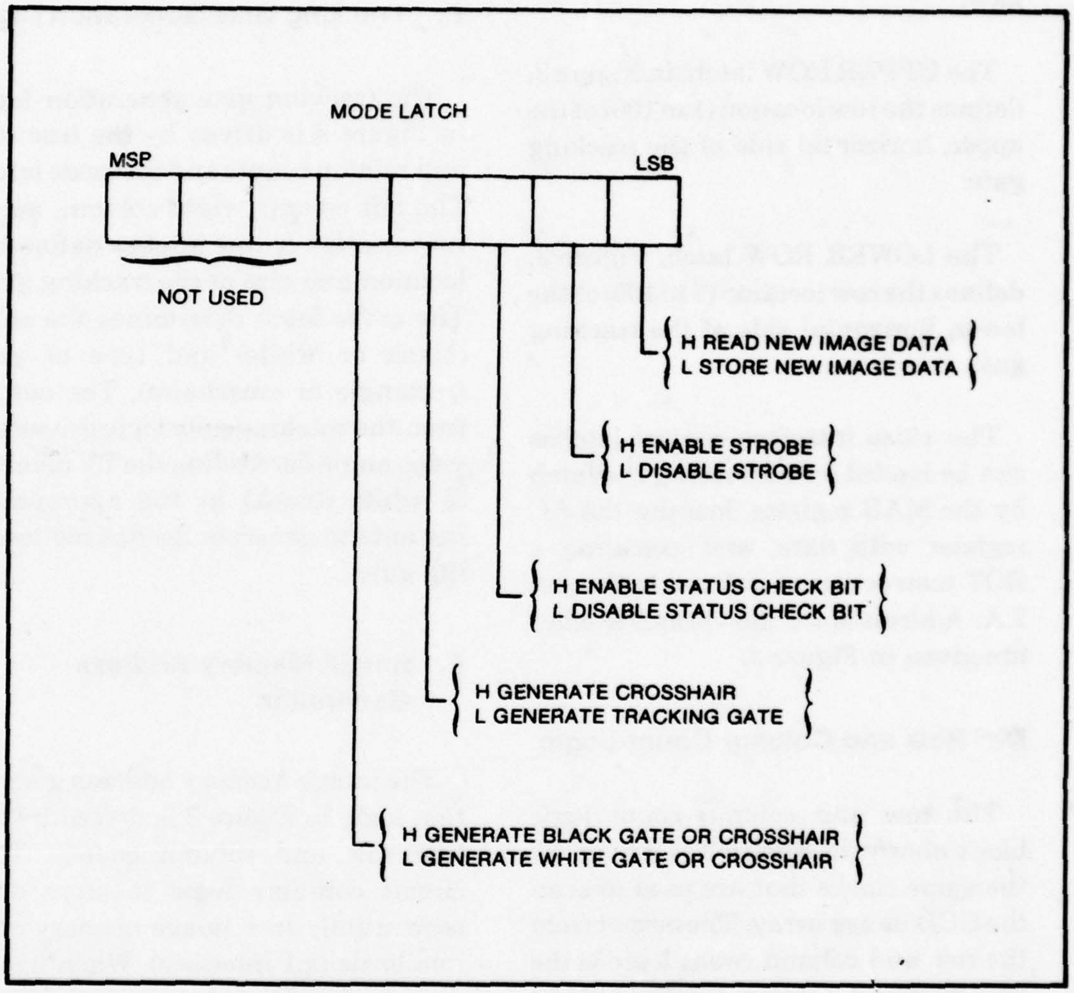


Figure 5. Bit assignments in the MODE latch.

of the left vertical side of the tracking gate.

The RIGHT COLUMN latch, Figure 3, defines the column location (1 to 100) of the right vertical side of the tracking gate.

The UPPER ROW latch, in Figure 3, defines the row location (1 to 100) of the upper, horizontal side of the tracking gate.

The LOWER ROW latch, Figure 3, defines the row location (1 to 100) of the lower, horizontal side of the tracking gate.

The video interface control latches can be loaded by addressing the latch by the MAR register, loading the AC register with data, and executing a ROT instruction as defined in Section 7.A. Addresses for the various latches are given in Figure 3.

D. Row and Column Count Logic

The row and column count logic block shown in Figure 3 is driven by the same clocks that are used to scan the CCD image array. The output from the row and column count logic is the true row and column location of the pixel being output from the TV camera. Since the camera uses 2:1 interlace, logic decoding and counters are necessary to produce the true row and column count. The row count ranges from 1 to 100 starting in the uppermost row. The column count also ranges from 1 to 100 but starts in the left most column.

E. Tracking Gate Generation Logic

The tracking gate generation logic in Figure 3 is driven by the true row and column counts and the mode latch. The left column, right column, upper row and lower row latches define the location and size of the tracking gate. The mode latch determines the color (black or white) and type of gate (rectangle or crosshairs). The output from the tracking gate logic drives the video amplifier feeding the TV monitor to white (black) at the appropriate instants to generate the desired tracking gate.

F. Image Memory Address Generation

The image memory address generation logic in Figure 3 is driven by the true row and column counts. This circuit contains logic to store data sequentially into image memory on a line basis (2:1 interlace). When image data is stored in this manner addressing of image data by the microprocessor becomes an easier task and need not take into account the effect of 2:1 interlace.

G. Conversion Control Logic

The conversion control logic in Figure 3 is driven by the same clocks used to scan the CCD array. When the frame signal is high, any strobes generated by the microprocessor will result in a new TV frame being stored into image memory, and the conversion control logic issues convert requests (REQ) to the analog to digital converter. The converter in turn issues an end of conversion signal (EOC) for each request (REQ). When data has stabilized and an EOC has been issued, data is written into image memory by bringing the memory write enable low (WE).

Since the conversion control logic uses the clocks which are also used to scan the CCD array, conversions take place as pixel data is shifted from the CCD and the row and column address of each new pixel is known for each conversion that takes place.

H. Image Memory

The image memory can store 10K x 4 bits of image data (all 100 x 100 elements of the array). The large memory is not required for tracker operation but is convenient for test purposes. Most correlation trackers would require no more than 32 x 32 = 1024 memory locations. The output of the memory is placed on the I bus of the microprocessor, but the data is meaningful only if addresses are being supplied from the microprocessor MAR bus. When a frame is being stored in image memory, the addresses are generated by the image memory address generation logic.

I. TV Monitor

The TV monitor shown in Figure 6 is used to monitor video and also the location and size of the tracking gate. In normal operation video is supplied directly to the monitor summed with the tracking gate signal. However, the monitor can also be used to display the stored image in image memory. When data is not being written into image memory, image data is available on the output of image memory. The output is used to drive a digital to analog converter (DAC). The output from the DAC is summed with TV sync and the tracking gate signal generating a composite video signal for the monitor. The image memory, address generation logic supplies addresses to image memory in synchronization with the clocks used scan to the CCD image array. The propagation delay form clocks to DAC output causes the video to be shifted relative to real time video, but still produces a very useful display. For example, the display can be used to determine the effects of preprocessing (gain and bias latches) on the video from the camera.

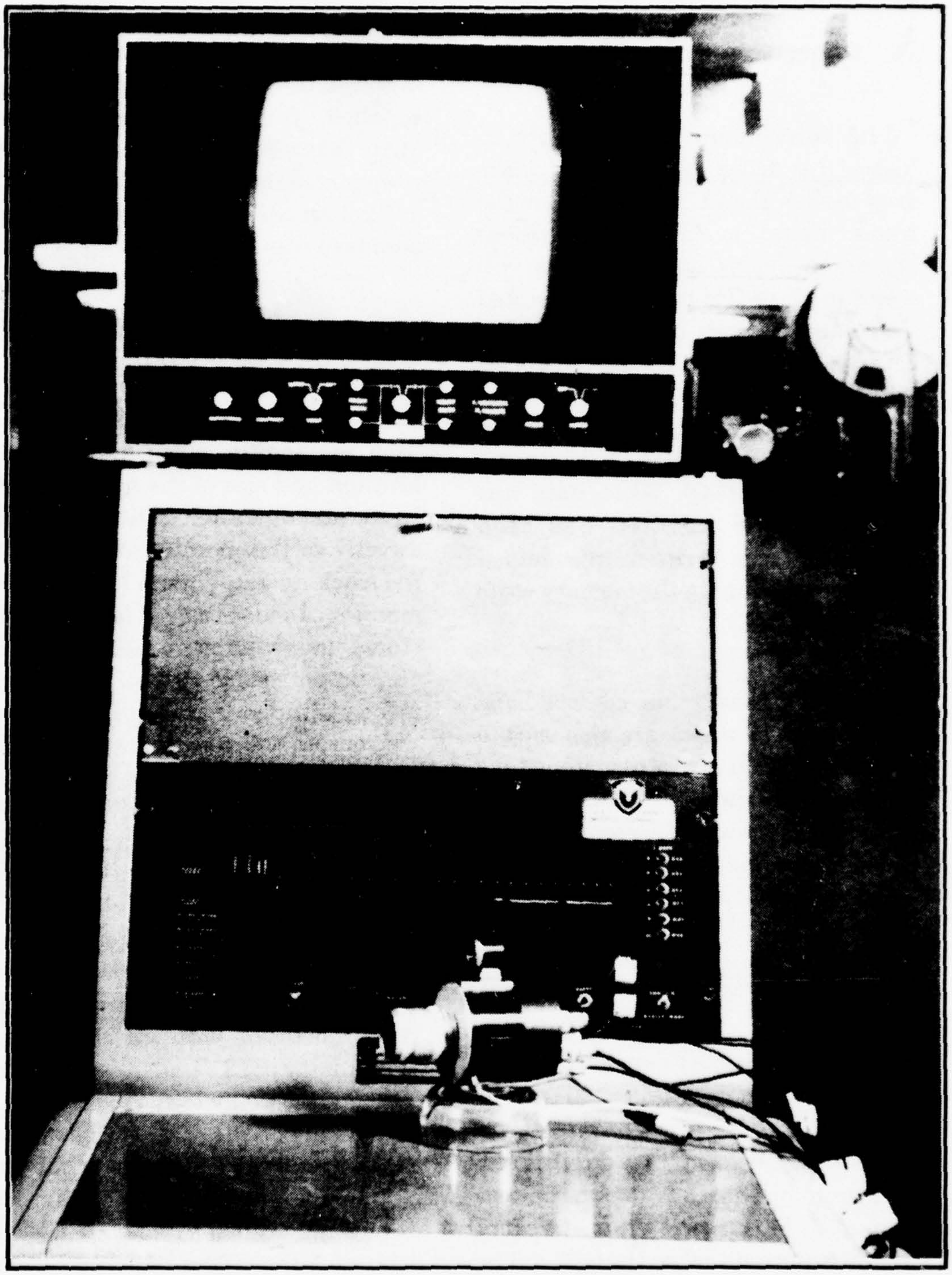


Figure 6. Photograph of tracker hardware.

4. MICROPROCESSOR AND TRACKER HARDWARE

A. Physical Description

Figure 6 is a photograph of the completed tracker and Figure 7 is a close up of the operator's panel. The thumb wheel switches shown in the figure drive address halt hardware. An address halt can be executed at any macroinstruction at any point within its microcode by setting addresses into the thumb wheel switches.

The selector switch shown at the left side of the panel drives multiplexer hardware which can display virtually any bus within the system. The column of switches to the right of the panel are software sense switches used for force branches in software.

The white switches to the lower right are the run (start the clock) and single switches (one clock pulse per depression). Since all memory can be read, modified and written into from the minicomputer there is no need to provide for these functions through front panel switches. The toggle switch located at the center, bottom portion of the panel selects one of two possible sources of video for the TV monitor. Video can be displayed directly from the camera or video generated from stored image data can be displayed. The toggle switch to the far right, bottom of the panel is used to take control of the data bus from the minicomputer. Finally, the small momentary push button switch located to the left of the run and single switches is used to reset hardware. Figures 8 and 9 show all of the circuit boards used in the tracker. Multiplexer boards are not shown. Some of the components on the memory boards are taken up by minicomputer interface circuit components. It should be noted that no attempt was made to minimize package size. Many improvements could be made along these lines. As an example the 2K x 48 bit RAM micro memory could be implemented with 1024 x 8 PROMs reducing the package count from 96 to 6 and the power requirements by a factor of 8.

B. Microprocessor and Tracker Architecture

The microprocessor shown in Figure 3 is shown in more detail in Figure 10. The microprocessor is the Intel 3000 used in a standard pipelined configuration. The loop counters shown are used extensively in image processing. The counters can be addressed and loaded, or addressed and incremented and tested on a single microcycle. The counters are used primarily to keep track of loop counts in the nested loop computations that occur in image processing (e.g. convolution and correlation computations).

Since counting, loading or testing can occur in parallel with microprocessor functions the overhead

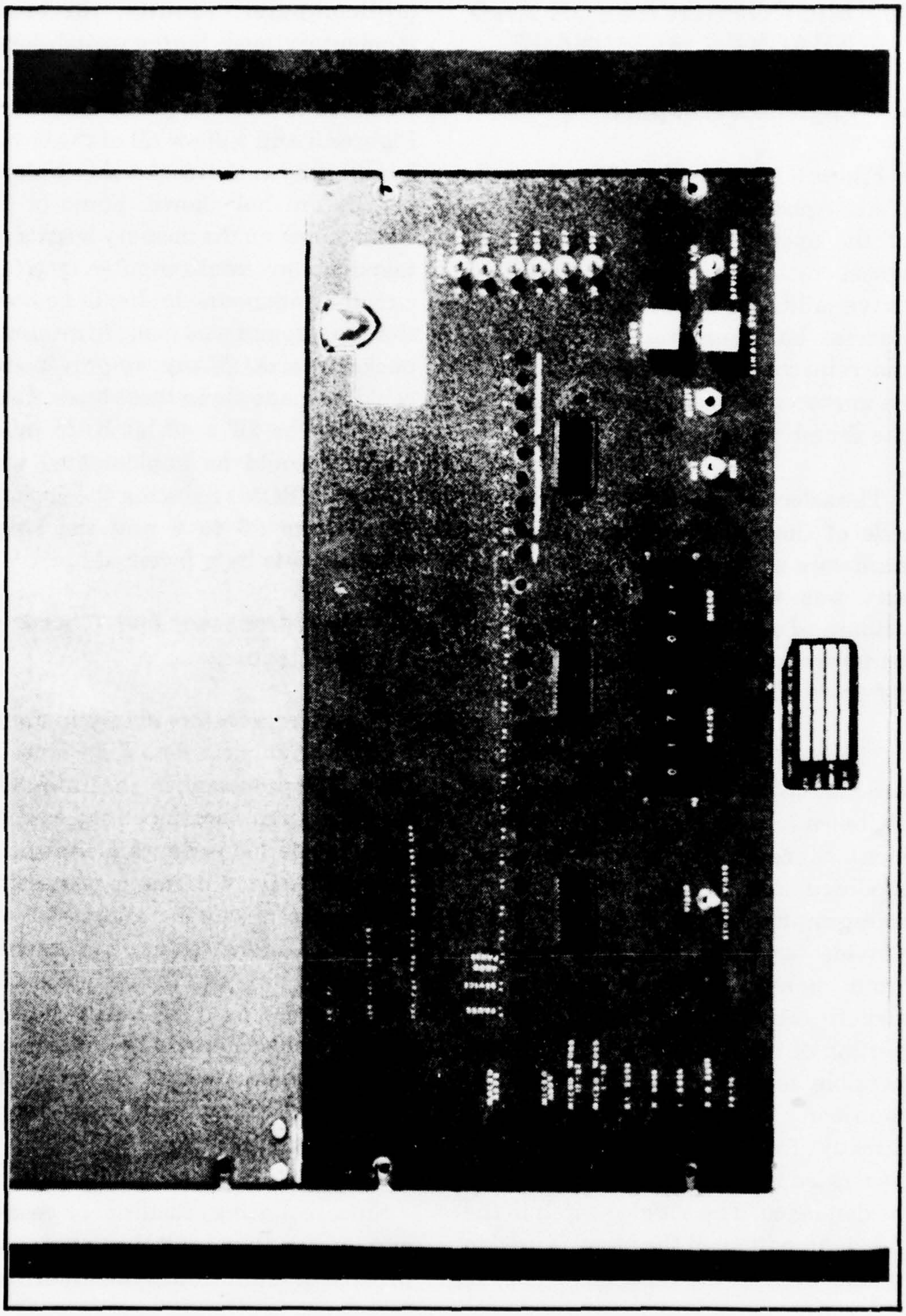
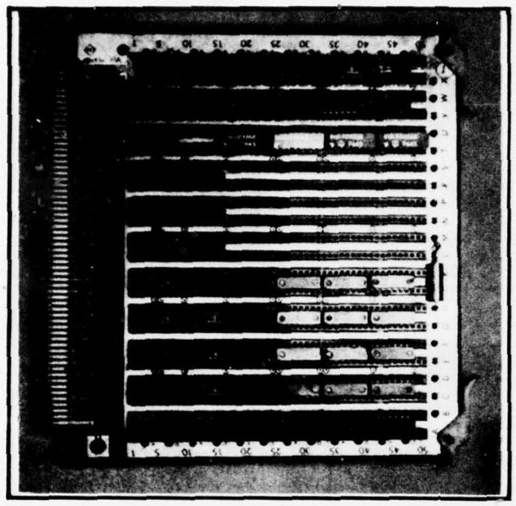
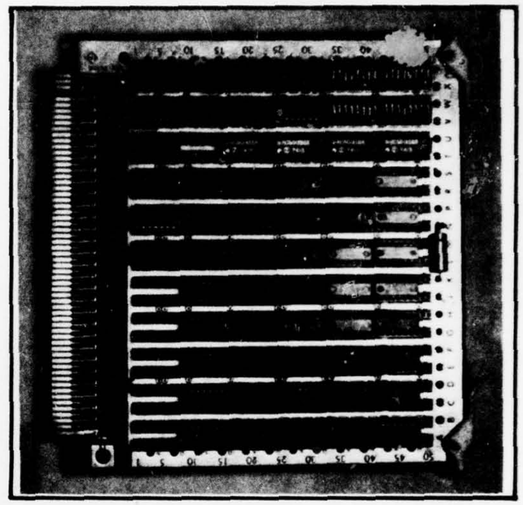


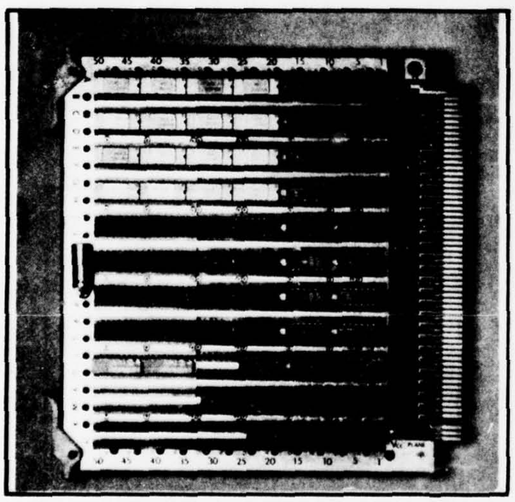
Figure 7. Photograph of operators panel.



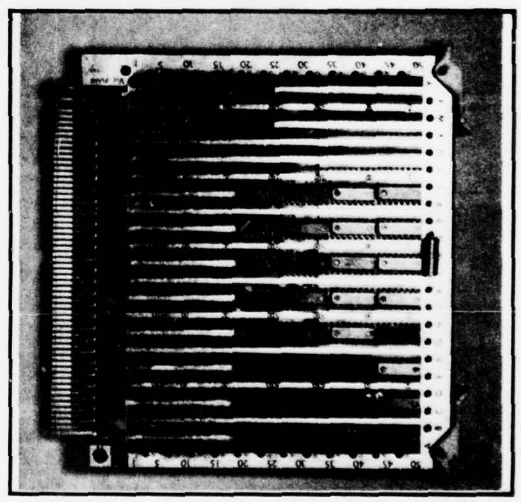
20 × 1K MACRO MEMORY



32 × 1K MICRO MEMORY

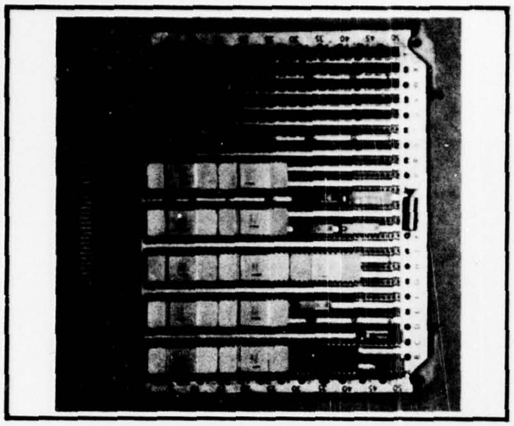


16 × 2K MACRO MEMORY

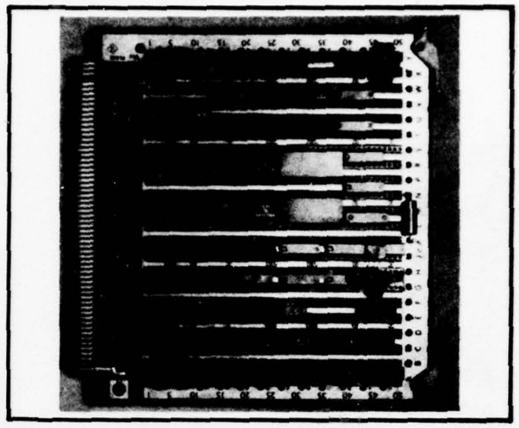


32 × 1K MICRO MEMORY

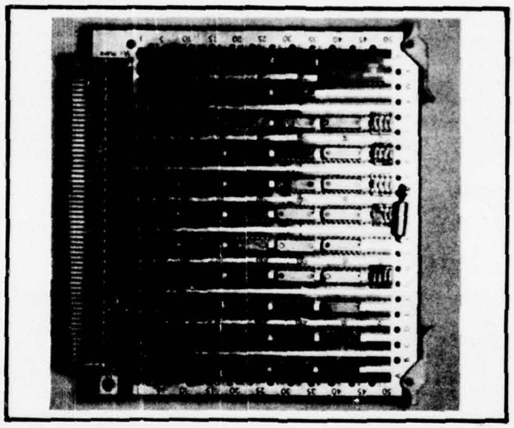
Figure 8. Macro and Micro memory boards.



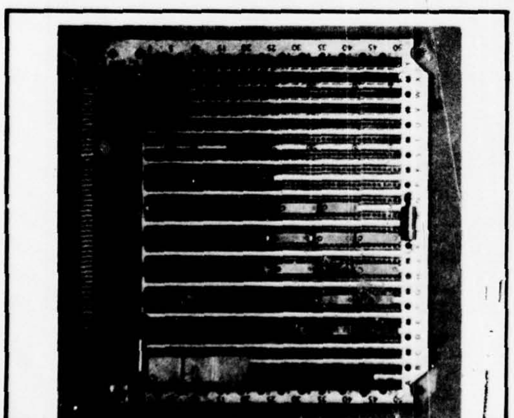
INTEL 3001 CPU



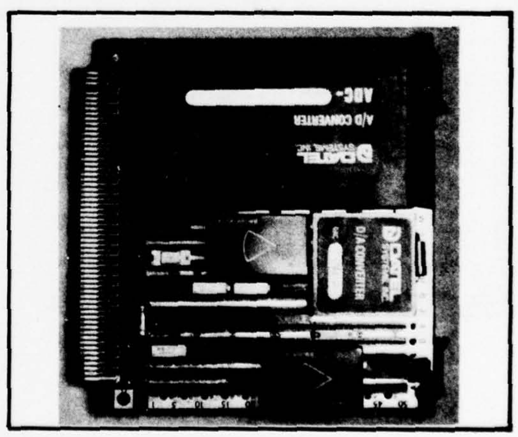
VIDEO CONTROL



10K × 4 IMAGE MEMORY



INTEL 3002



VIDEO FRONT END

Figure 9. Microprocessor, image memory, and video front end boards.

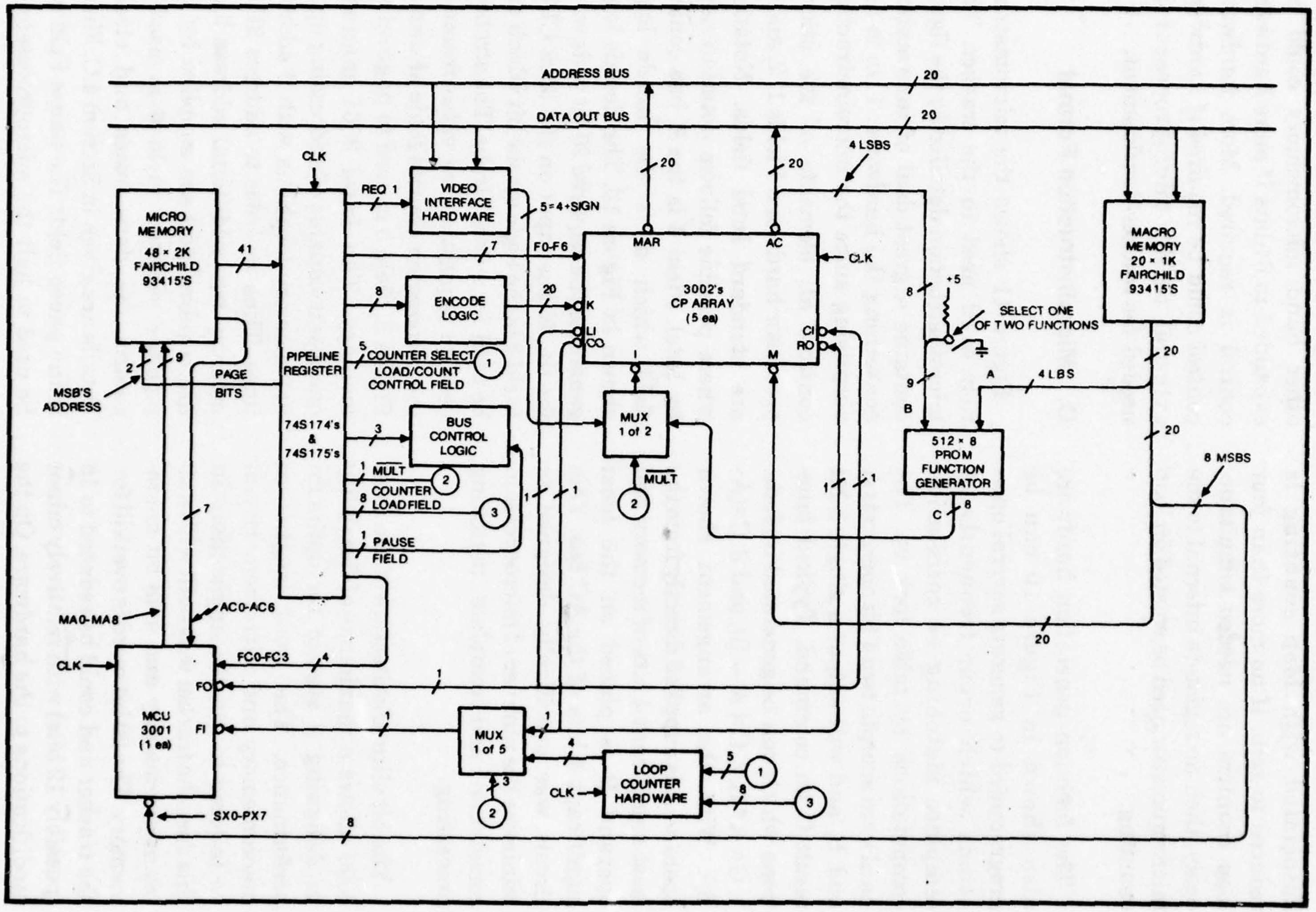


Figure 10. Block diagram of microprocessor and tracker architecture.

associated with loop counting is reduced to zero. If no more than four loop counters are needed simultaneously, then no registers internal to the microprocessor need be wasted on loop counting.

The function generation hardware also shown in Figure 10 can be programmed to generate several operations which occur frequently in template matching or correlation computations by table look up. The hardware accepts two 4 bit operands A and B, and will output a single 8 bit result C on command. Typical functions which can be generated include: 1. $C = A * B$, 2. $C = |A - B|$, and 3. $C = (A - B)^2$. With the arrangement shown operand A is supplied directly from the least significant 4 bits of memory and operand B is placed on the least significant 4 bits of the AC bus. This circuit was specifically designed to minimize the number of microcycles in correlation and template matching processing.

The bit slice architecture of the Intel 3000 allows a maximum of flexibility in designing a system for optimum performance. The word lengths of macromemory and micromemory can be tailored to suit the application. In this case the tracker was built with a 20 bit macromemory and a 48 bit micromemory. The 20 bit word is overkill for the tracker and could be reduced to 16 (possibly 12 bits) with relatively minor modifications to the hardware. On the other hand micromemory could be expanded to 64 bits if more hardware control is required. More hardware control could be required if hardware external to the microprocessor is needed for tracker development.

C. Microinstruction Format

Figure 11 shows the microinstruction word used in the tracker. The microinstruction detailed in the figure contains a great deal of information concerning the hardware. This is not surprising since the microinstruction controls all elements of the microprocessor hardware. Fields 1, 2, and 3 are standard Intel fields. Notation where possible follows notation used by Intel. Field 4 is the K bus control field which drive the decode logic shown in Figure 10. The decode logic generates the required 20 bit wide word for the K bus input on the 3002 CPU. Field 8 is the logic strobe bit which can be used for a data strobe. The strobe is used to request a new video frame in the conversion control logic shown in Figure 3. Field 5 is used to page micromemory. The Intel 3001 microsequencer is capable of addressing up to 512 memory locations with 9 address lines. Thus in order to address 2K of memory two additional address lines are required and are supplied by the pipeline register. Field 6 is used to control the data buses and clocks. Details are given in Section 4.C. Field 7 is the pause field. The pause field can be used to halt the microprocessor by

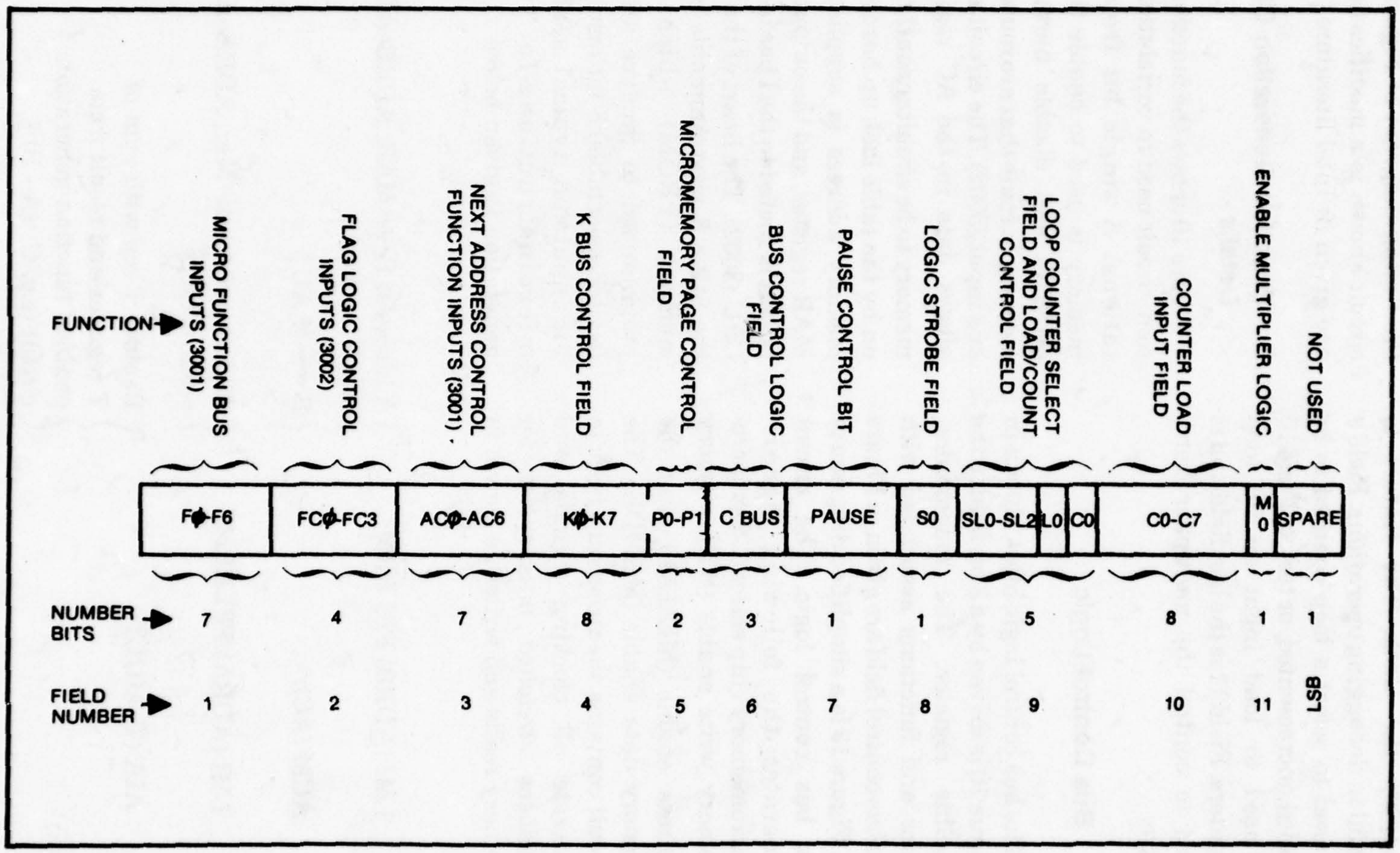


Figure 11. Microinstruction format (48 bits).

stopping the clock. The feature is useful in debugging operations. Field 9 is used to select a loop counted to be loaded, incremented, or tested. Field 10 is used to load input on all loop counters. Field 11 is the last field and is used to control the multiplier hardware.

D. Bus Control Logic

The bus control logic block shown in Figure 10 is driven by a 3 bit field in the pipeline register. The mnemonics, codes and functions associated with the bus control field are given in Figure 12. Figure 13 is a simplified diagram of the bus control logic. The circuit generates the following signals: macromemory chip select (CS), macromemory write enable (WE), memory address enable (MEMAE), and the memory data enable (MEMDE). The circuit contains the necessary logic to generate all clocking signals, and produces stretched microcycles for memory reads and writes as shown in the timing diagram of Figure 14. The circuit shown is a modification of the one given in Intel literature [1].

E. Function Generation Circuit Details

Figure 10 shows the function generation circuit used in correlation computations. A single bit from micromemory is used to enable the circuit (MULT) and disable normal video memory outputs that are routed to the I bus input (3002). The circuit as shown allows data in the AC register and memory to be simultaneously operated on by the table look up hardware. The memory address is supplied by the MAR register, and the output from the circuit is routed to the I bus input of the CPU (3002). The heart of the circuit is the 512 × 8 programmable read only memory (PROM) which can be programmed to produce any of two (switch selectable) 8 bit results from 4 bit inputs. A typical sequence of microinstructions for function generation is given below.

LMI (ADRB) FFI RRM;	{Address of B → MAR, ADRB=ADRB+1, read - memory}
ACM (AC);	{B → AC}
LMI (ADRA) FFI RRM;	{Address of A → Mar, ADRA=ADRA+ read memory}
AIA (T) MULT;	{Replace T reg with sum of T register and result from enabled function generation circuit (e.g. C =(A - B) 2}

MNEMONIC	CODE	FUNCTION
NMO	000	NO BUS OPERATION
INH	001	INHIBIT CPU CLOCK
RMW	010	READ MODIFY WRITE
CNB	011	CPU NEEDS BUS
RIN	100	REQUEST INPUT FROM EXTERNAL DEVICE
ROT	101	REQUEST OUTPUT TO EXTERNAL DEVICE
RRM	110	REQUEST READ MEMORY
RWM	111	REQUEST WRITE MEMORY

Figure 12. Bus control mnemonics.

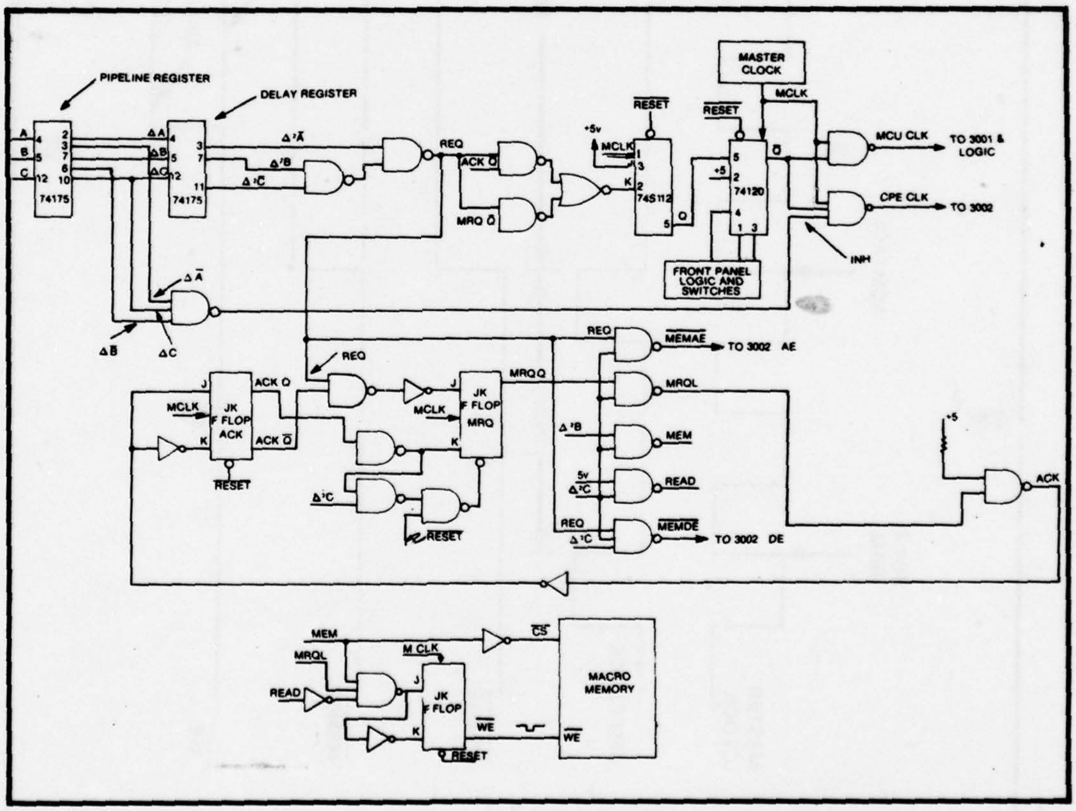


Figure 13. Simplified bus control logic diagram.

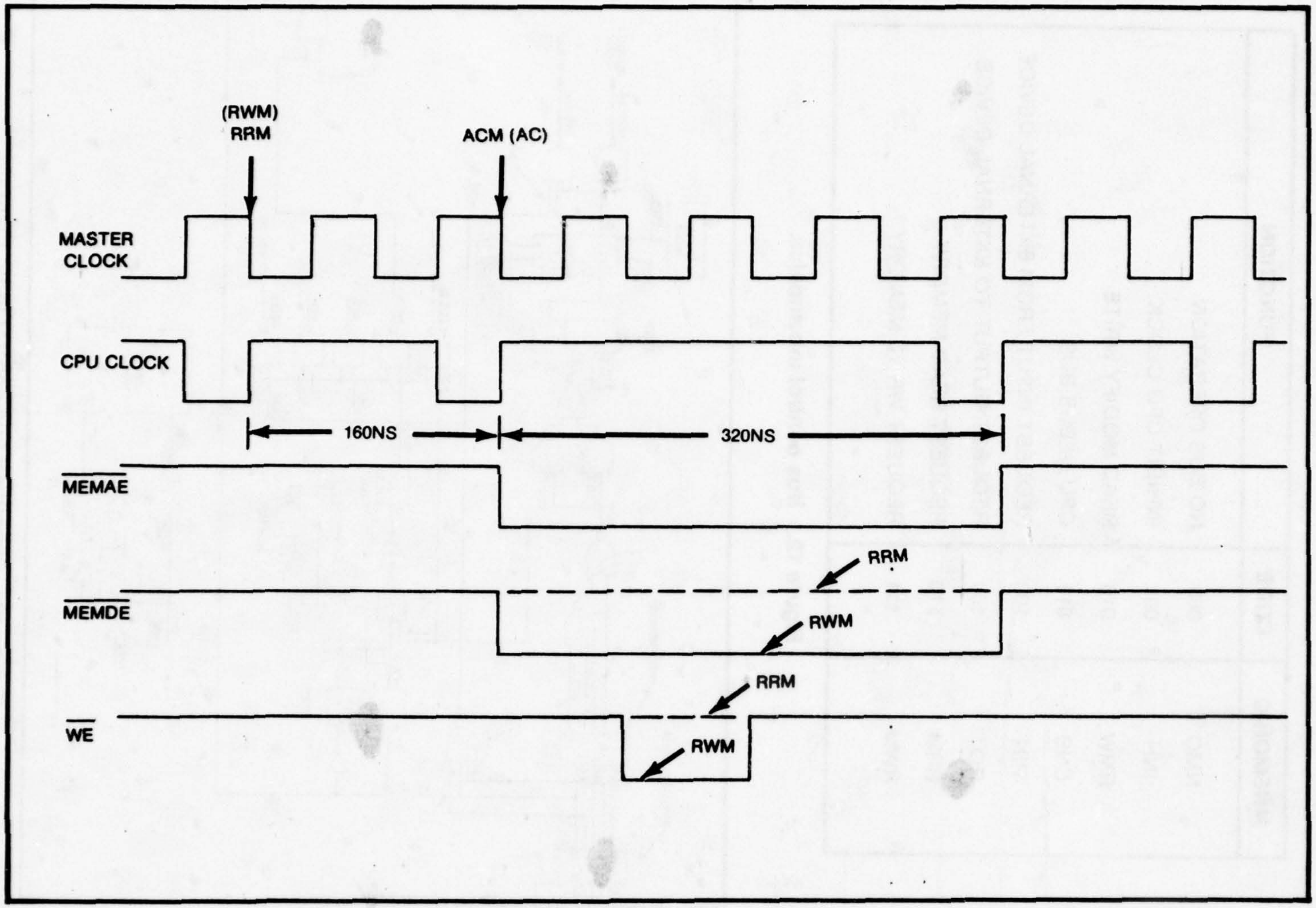


Figure 14. Bus control timing.

Hardware timing is shown in Figure 15 and relates directly to the example given above. The logic signal MEMAE is the macromemory address enable generated by the bus control logic, and the logic signal MULT is asserted (active low) by a bit in the micro-instruction word (MULT). The stretched cycles for slow memory are evident in the diagram.

F. Loop Counting Circuit Details

Figure 16 shows the details of loop counting hardware. There are four counters shown, thus, loop counts for four computation loops may be maintained simultaneously. The counters have general utility and are especially useful when dealing with nested loops appearing in image processing. The counters may be loaded (or incremented and tested) simultaneously with CPU operations of the Intel 3002. The field mnemonics and associated hardware control word are shown in Figure 16. In practice a loop counter is initialized by loading a selected counter with the ones complemented of the desired loop count. The counter can be incremented and tested as required by testing for a carry from the counter. When counters are not being tested or loaded, the source of the Intel 3001 FI input is the carry out (CO) from the Intel 3002.

An example segment of code is given below showing how a counter is loaded and then tested by the JFL instruction of the Intel 3001.

The clock signals shown in Figure 16 are delayed ¼ of a microcycle from the MCU clock so that loading (or incrementation and testing) can be accomplished in a single microcycle.

```
        ILR(R0)    LCNTI M5;           { R0 → AC, load counter
                                             one with 5 }

LOOP:   ALR(AC)    ITCNTI;             { shift AC register one time to left,
                                         increment and test counter one }

        ADR(R1)    JFL(LOOP, CONT);    { R1 = R1 + AC, if counter
                                         had a carry branch to
                                         CONT else, go to LOOP }

CONT:   SDR(R3)
```

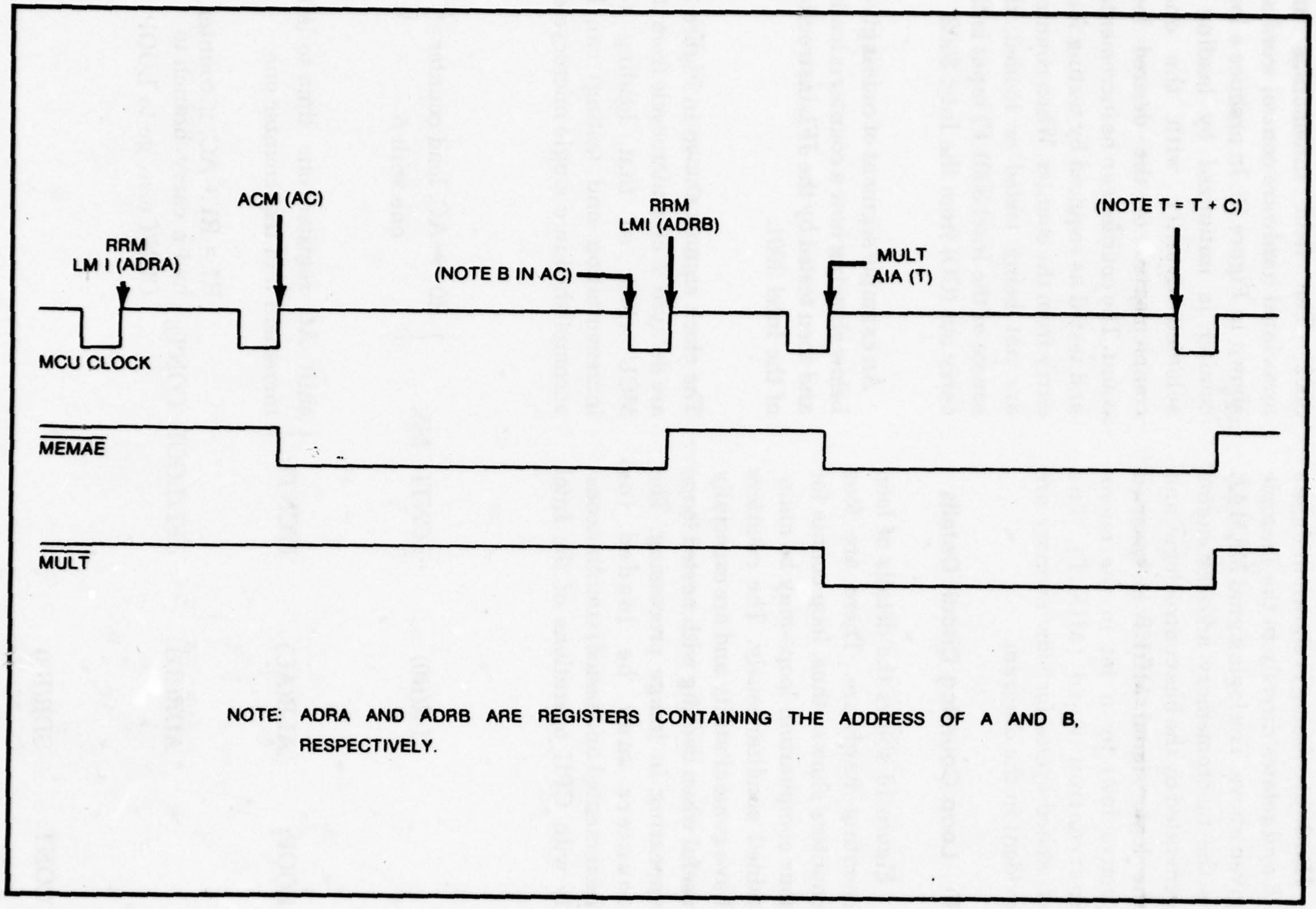
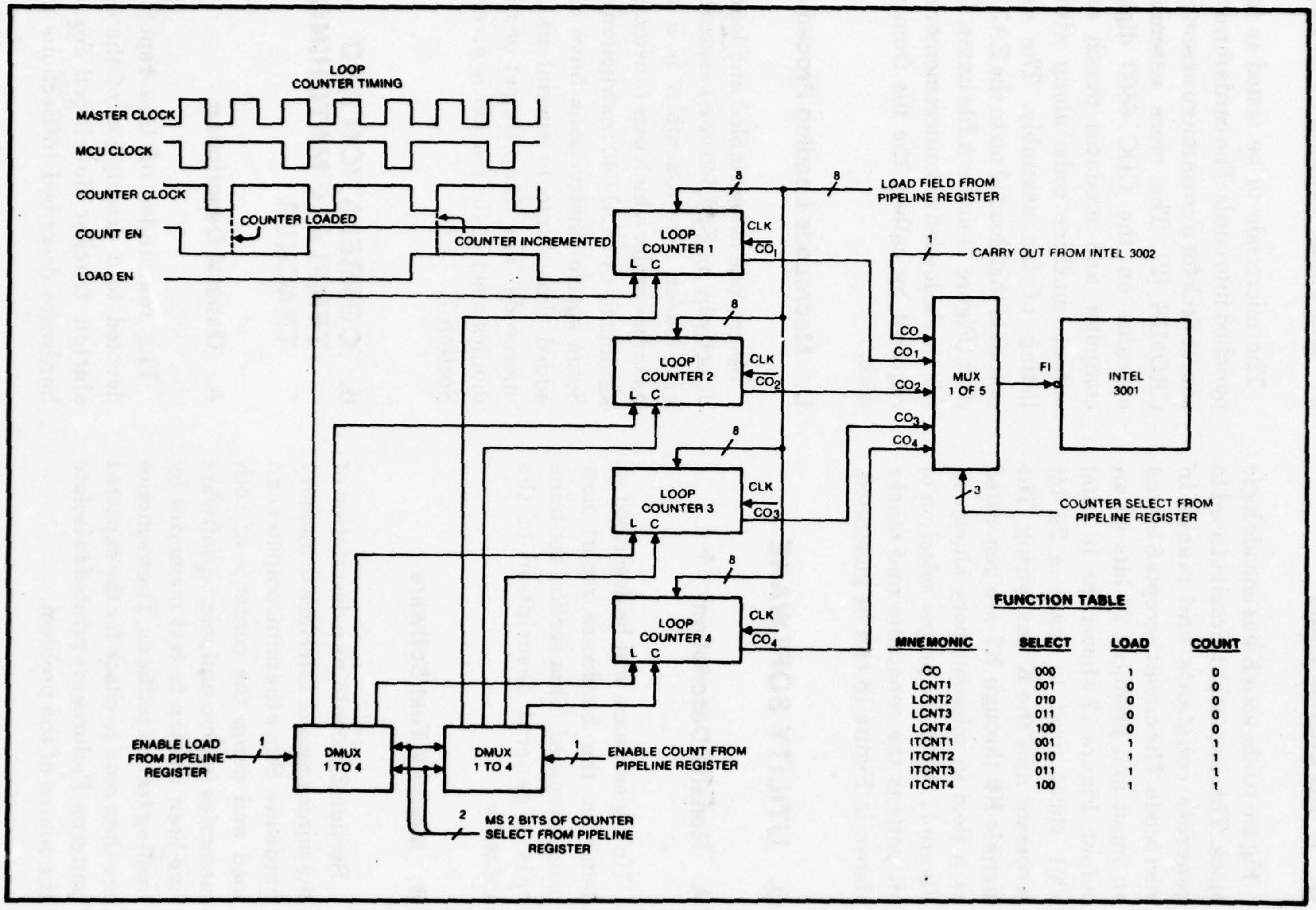


Figure 15. Function generation timing.



FUNCTION TABLE

MNEMONIC	SELECT	LOAD	COUNT
CO	000	1	0
LCNT1	001	0	0
LCNT2	010	0	0
LCNT3	011	0	0
LCNT4	100	0	0
ITCNT1	001	1	1
ITCNT2	010	1	1
ITCNT3	011	1	1
ITCNT4	100	1	1

Figure 16. Details of loop counter hardware.

G. K Bus Encode Logic

Figure 10 shows a K bus encode logic block. This important circuit is used to generate constants and masks in microcode. The circuit accepts 8 bits as an input and produces 20 bits as an output. Figure 17 shows the 10 Intel 3001 slices used to make a 20 bit processor and the K bus wiring. The signals K0 through K7 are generated as a field in micromemory shown in Figure 11. With the proper selection of bit pattens the constants and masks shown in Figure 18 can be generated.

5. UTILITY SOFTWARE

A. General Description

Utility software will be described in detail in the hardware report now being prepared. This section contains only a general description of the software.

B. Memory Test Software

Section 2.B contains a description of the minicomputer interface to memory hardware. Since the minicomputer can load and dump the contents of both memories (macro and micro), software has been written to test memories by loading test bit patterns. The memories are then read to check for the expected patterns. Failures are printed to isolate the source of the problem.

C. Microcode Loading Procedure

The microcode to be tested is first punched into cards. The cards form the source deck for a cross micro assembler CROMIS [2]. The cross assembler operates on the CDC 6600 digital computer and produces punch card output (machine code) along with a listing of the assembly. The card output is then loaded on to the EAI 640 disk (Figure 1) under a file name. The file can be loaded into micromemory as required by calling the file from the disk.

D. Macrocode Loading Procedure

Macrocode is assembled and loaded directly by an EAI 640 cross assembler and loader. The assembler is a two pass assembler which uses mnemonics similar to the EAI 640 instruction set. Some special instructions have been added that relate to special tracker microcode. The complete set of mnemonics used in the tracker is given in Section 7.

6. CORRELATION AND TEMPLATE MATCHING TRACKER

A. General Description

The remainder of this report is devoted to a description of the correlation tracker developed for the hardware described in Sections 2, 3,

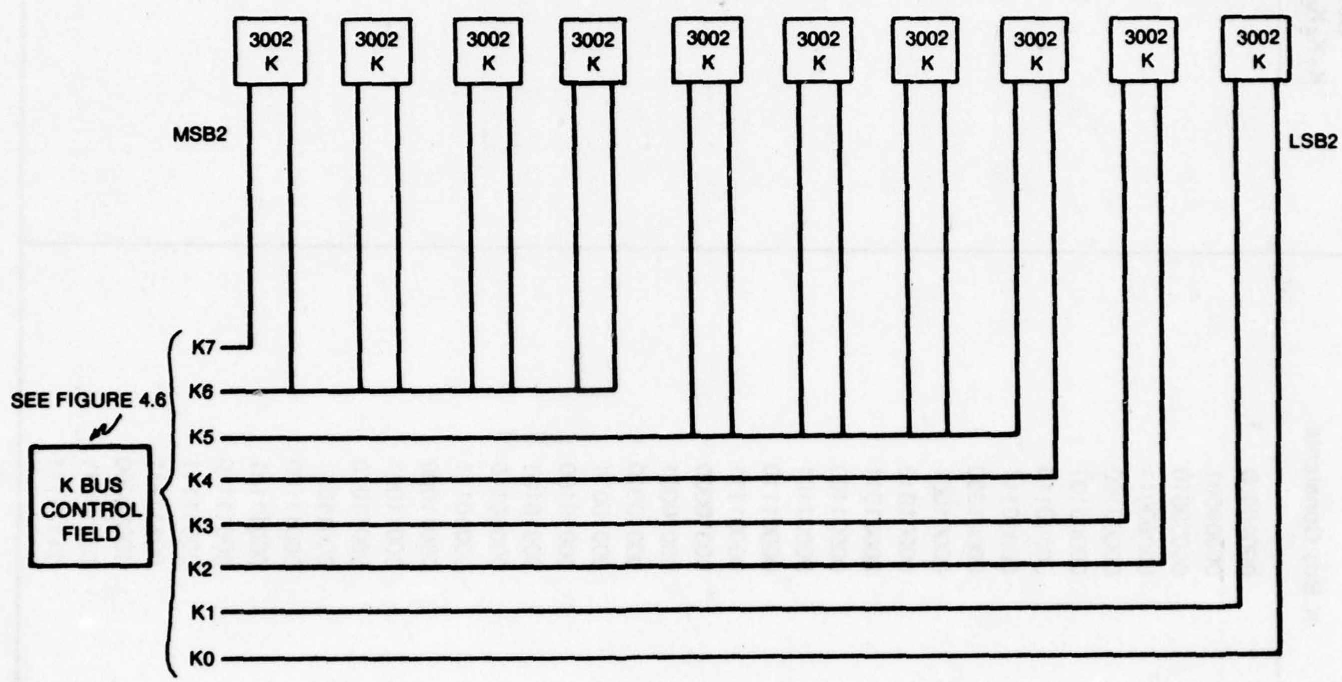


Figure 17. K bus wiring for logic encoding.

K Bus Constants	K Bus Control Field $K_7K_6K_5K_4K_3K_2K_1K_0$
00000000	00000
00000001	00001
00000010	00002
00000011	00003
00000100	00004
00000101	00005
00000110	00006
00000111	00007
00001000	00008
00001001	00009
00001010	0000A
00001011	0000B
00001100	0000C
00001101	0000D
00001110	0000E
00001111	0000F
00010000	00010
00010001	00011
00010010	00012
00010011	00013
00010100	00014
00010101	00015
00010110	00016
00010111	00017
00011000	00018
00011001	00019
00011010	0001A
00011011	0001B
00011100	0001C
00011101	0001D
00011110	0001E
00011111	0001F
00111111	01FFF
10000000	80000
01111111	7FFFF
11111111	FFFFF

Figure 18. Bits patterns required for K bus constants and masks generation.

and 4. The tracker is unique in that it operates in real time and uses almost 100% software to execute algorithms.

Appendix A contains a listing of the tracker macrocode. The macrocode contains EAI 640 instructions (emulation) along with some special instructions associated with the tracker. The EAI 640 instructions are given in Section 6.B. The special instructions, described in detail in Section 7, represent the bulk of the tracker software. The software described in these sections is written on the microcode level to achieve maximum efficiency. A complete listing can be found in Appendix B.

B. Correlation and Template Matching Algorithms

The complete tracker will be described in Section 6.C. This section identifies several correlation and template matching algorithms selected for study. Though several algorithms were tested, all operate on two image subarrays as defined in Figure 19. The 17 × 17 image array I(I,J) is part of the 100 × 100 image produced by the camera. The smaller 9 × 9 reference array R(I,J), bounded by the tracking gate, contains a high percentage of target pixels relative to background pixels.

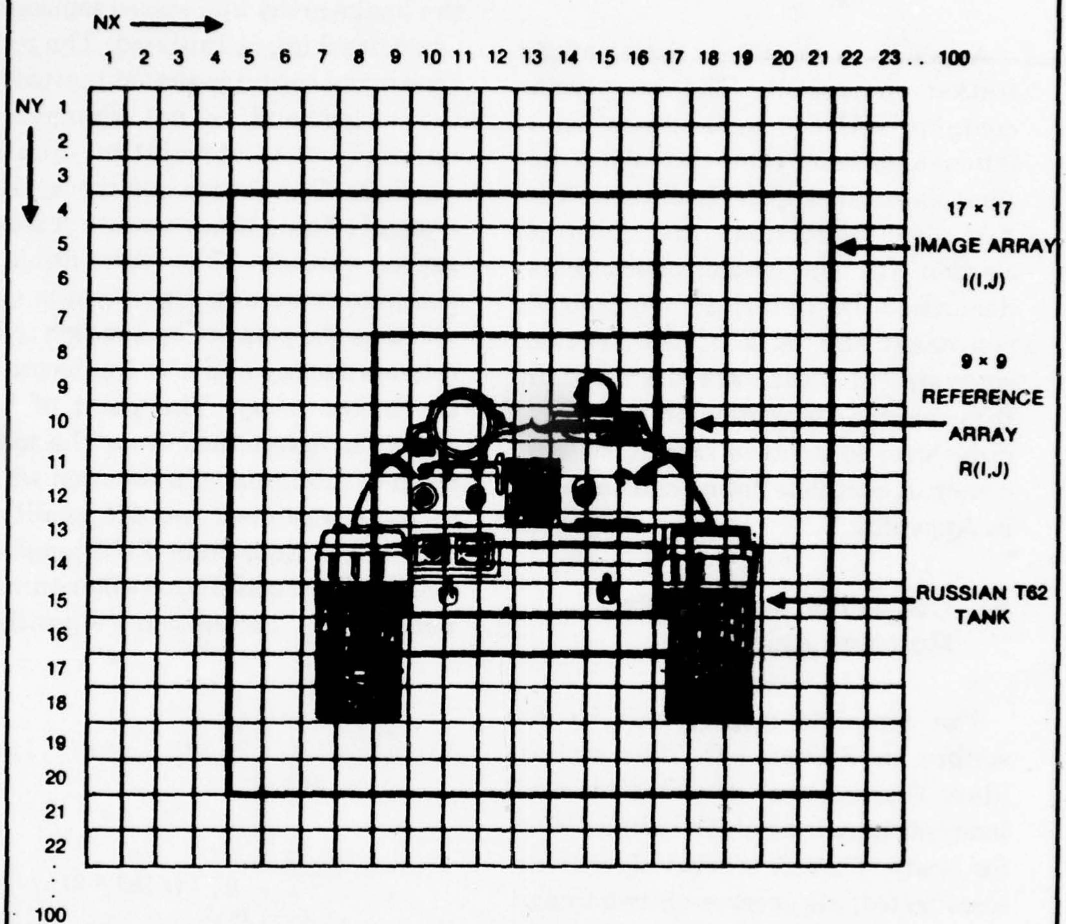
The image array is always centered at the current track point (NX, NY). The image array is read from the 10K video memory 123 times a second during the vertical blanking interval. The reference array is obtained from the image array and stored separately when tracking is initiated. The reference array contains the "signature" of the target and is not changed or updated until correlation quality degrades. Correlation quality degrades typically due to range closure or target aspect changes. The correlation or template matching algorithm is used to locate the point of best match of the reference array and a 9 × 9 subarray of the image array. The point of best match is determined from the maximum or minimum of a function which in some way measures the quality of the match. Equations 6.1 through 6.3 define all the correlation algorithms or normalized correlation algorithms

$$\{C(M,\bar{N})\} = \sum_{i=1}^{9} \sum_{j=1}^{9} I(K,L) * R(I,J) \qquad 6.1$$

$$\{C(M,N)\} = \frac{\sum_{I=1}^{9} \sum_{J=1}^{9} I(K,L) * R(I,J)}{\sum_{I=1}^{9} \sum_{J=1}^{9} I^2(K,L) \sum_{I=1}^{9} \sum_{J=1}^{9} R^2(I,J)} \qquad 6.2$$

$$\{C(M,N)\} = \sum_{I=1}^{9} \sum_{J=1}^{9} (I(K,L)-\bar{I}) * (R(I,J)-\bar{R}) \qquad 6.3$$

$$\{E(M,N)\} = \sum_{I=1}^{9} \sum_{J=1}^{9} |I(K,L)-R(I,J)| \qquad 6.4$$



NOTE
1. Track point illustrated is NX = 13 and NY = 12
2. Tracking gate location is indicated on the TV monitor by intensified (or darkened) pixels forming a perimeter around the 9 × 9 array, one pixel wide.
3. Only a portion of the 100 × 100 CCD array is shown.

Figure 19. Definition of image and reference arrays.

$$\left\{E(M,N)\right\} = \sum_{I=1}^{9} \sum_{J=1}^{9} (I(K,L)-R(I,J))^2 \qquad 6.5$$

where $K=I+M-1$, $L=J+N-1$, $1 \leq M \leq 9$,

$1 \leq N \leq 9$ and where

$$\overline{R} = \frac{1}{81} \sum_{I=1}^{9} \sum_{J=1}^{9} R(I,J)$$

$$\overline{I} = \frac{1}{81} \sum_{I=1}^{9} \sum_{J=1}^{9} I(I,J)$$

tested from imagery from the Fairchild 201 CCD TV. As a preliminary test, a 35MM slide representative of the IR signature of a target was projected on a screen. The image data obtained in this manner is approximately representative of the image data from a camera operating in the IR spectrum.

Figure 20 is a photograph of the TV monitor showing the projected image of such a slide. The bright areas correspond to the hottest areas in target, which in this case, are the tracks and boggey wheels.

The image and reference arrays shown in Figure 21 were obtained from the Fairchild 201 CCD TV and microprocessor hardware. The reference array is the center 9 × 9 of the 17 × 17 image array. The arrays were used to test the candidate tracking algorithms listed in Equations 6.1 through 6.5. Each algorithms results in a 9 × 9 array C (M,N) or E(M,N).

The correlation value and the side lobes of the match point give an indication of the robustness of the algorithm. Results from Equations 6.1 through 6.5 are presented in Figures 22 through 26 respectively. It is clear from the testing that scene brightness variation causes false peaks in the unnormalized correlation result. The results were surprising in view of the fact the shifting distance is so small. This test is representative of other results which also indicate the unnormalized correlation is not adequate as a tracking algorithm. Two normalization schemes were tried on the same data with improved results. In the case of normalization by norms the Cauchy Schwarz inequality guarantees a result less than one and less sensitivity to scene brightness variation. The result is given in Figure 23. Normalization by means as in Equation 6.3 gives similar results as shown in Figure 24. From an implementation point of view normalization by means is more difficult. As shown in Equation 6.3 the mean of the image 9 × 9 subarray and reference arrays must be known before the computation takes place. On the other hand, though a division is required, the normalization by norms for each correlation result can take place after the standard

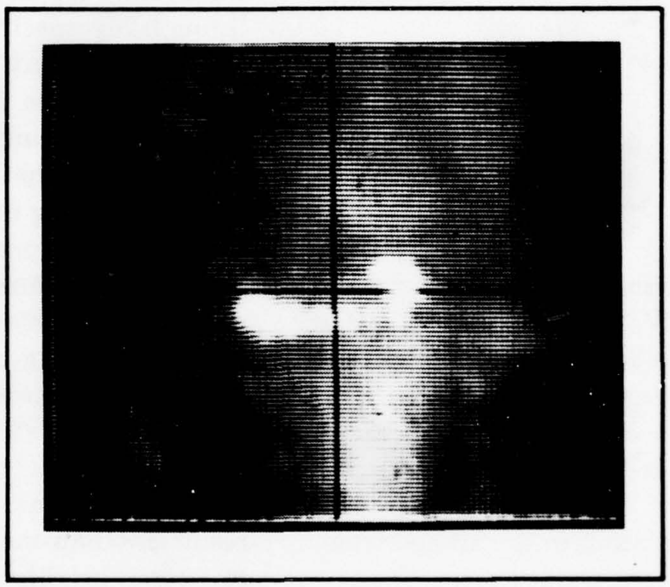


Figure 20. Video IR image of the side view of an M48 tank.

IMAGE ARRAY

3	2	2	2	4	3	3	3	3	2	2	1	3	3	4	5	6
3	3	3	3	4	3	3	4	5	3	4	2	3	4	7	6	7
3	1	4	4	3	3	3	3	5	4	4	3	0	5	4	5	8
3	2	4	5	4	4	4	3	4	4	3	3	5	4	6	7	10
4	3	4	4	4	3	3	4	4	3	3	2	7	5	7	8	10
5	5	5	5	6	5	5	5	6	6	5	5	7	7	9	9	11
5	4	4	4	5	5	5	6	6	6	6	7	7	6	8	7	9
7	7	6	6	6	5	6	6	8	6	8	7	7	7	7	7	8
8	6	7	7	7	5	6	5	6	6	6	5	6	6	6	7	7
9	8	9	9	8	7	8	7	7	6	7	7	7	8	7	7	8
10	9	9	9	7	7	7	7	8	6	8	8	8	8	8	8	9
13	11	12	12	9	9	9	12	12	10	10	12	13	11	10	11	12
13	12	14	13	12	12	12	14	15	11	10	13	13	13	10	10	12
14	15	15	15	13	13	15	15	15	11	12	12	15	13	12	11	11
13	15	15	15	14	13	14	15	15	11	11	12	13	11	11	10	10
13	15	15	15	13	12	13	13	14	10	9	10	12	11	10	9	11
11	12	12	11	9	8	9	9	9	7	7	7	8	8	8	7	8

REFERENCE ARRAY

4	3	3	4	4	3	3	2	7
6	5	5	5	6	6	5	5	7
5	5	5	6	6	6	6	7	7
6	5	6	6	8	6	8	7	7
7	5	6	5	6	6	6	5	6
8	7	8	7	7	6	7	7	7
7	7	7	7	8	6	8	8	8
9	9	9	12	12	10	10	12	13
12	12	12	14	15	11	10	13	13

Figure 21. Image and reference arrays used for testing. (See Equation 6.1)

CORRELATION UNNORMALIZED

2834	2797	2854	2822	2870	2895	3054	3195	3464
3202	3130	3170	3116	3138	3170	3311	3410	3668
3555	3440	3454	3410	3419	3474	3589	3705	3948
4152	4012	4007	3995	4066	4088	4204	4331	4537
4917	4770	4738	4733	4837	4826	4877	4936	5074
5668	5505	5434	5408	5455	5463	5465	5437	5498
6273	6088	5981	5917	5916	5882	5850	5779	5745
6700	6483	6321	6190	6156	6095	6026	5914	5863
6660	6430	6226	6063	5997	5947	5884	5761	5708

Figure 22. Unnormalized correlation. (See Equation 6.3)

CORRELATION NORMALIZED BY NORMS

.97687	.96886	.97129	.96364	.95890	.95647	.94968	.93949	.92907
.97082	.96125	.96326	.95564	.95332	.95268	.94723	.93532	.92957
.97633	.96632	.96602	.96342	.96134	.96506	.95976	.95664	.94994
.97528	.96819	.96875	.97220	.98025	.97359	.97180	.97219	.97053
.98507	.97989	.97953	.98524	1.00000	.98627	.98097	.97792	.98027
.98026	.97490	.97294	.97858	.98452	.97798	.97121	.96638	.97101
.97716	.97017	.96792	.97093	.97375	.96714	.96245	.96050	.96371
.96379	.95476	.95165	.95164	.95525	.94816	.94251	.93991	.94517
.93266	.92343	.91831	.91735	.91971	.91585	.91224	.90977	.91674

Figure 23. Correlation normalized by norms.

CORRELATION NORMALIZED BY MEANS

270.00000	240.00000	241.00000	237.00000	250.00000	247.00000	252.00000	214.00000	224.00000
309.00000	258.00000	231.00000	230.00000	238.00000	217.00000	211.00000	170.00000	183.00000
350.00000	284.00000	263.00000	254.00000	256.00000	269.00000	258.00000	248.00000	235.00000
439.00000	376.00000	364.00000	380.00000	409.00000	389.00000	379.00000	380.00000	365.00000
598.00000	549.00000	531.00000	554.00000	616.00000	556.00000	509.00000	477.00000	475.00000
652.00000	601.00000	565.00000	574.00000	600.00000	566.00000	505.00000	449.00000	440.00000
665.00000	613.00000	569.00000	561.00000	567.00000	519.00000	466.00000	423.00000	396.00000
521.00000	465.00000	422.00000	421.00000	436.00000	382.00000	327.00000	285.00000	276.00000
250.00000	202.00000	159.00000	150.00000	179.00000	150.00000	115.00000	83.00000	93.00000

Figure 24. Correlation normalized by means. (See Equation 6.4)

SUM OF ABSOLUTE DIFFERENCES

231	232	224	228	223	223	207	202	195
186	197	190	199	187	198	187	185	180
150	161	158	169	166	158	156	158	158
112	123	124	126	108	112	116	116	117
84	98	92	70	0	67	87	100	96
149	147	144	123	112	132	149	159	149
207	206	195	179	170	188	197	191	192
284	281	270	252	227	242	252	248	236
351	343	334	322	311	312	316	309	294

Figure 25. Sum of absolute differences template matching. (See Equation 6.5)

METRIC

909	966	914	966	949	941	867	838	783
682	769	736	803	801	786	741	765	720
468	577	572	607	614	568	550	528	512
280	363	360	338	262	306	298	278	281
154	196	198	142	0	135	193	232	228
413	419	418	335	274	362	453	507	469
811	802	769	681	636	720	775	763	694
1428	1403	1316	1204	1111	1190	1236	1194	1066
2059	2001	1888	1732	1633	1660	1670	1605	1436

Figure 26. Sum of square differences template matching.

correlation. The norm of the image array may be conveniently computed at the same time as the correlation result. Two template matching algorithms [3] shown in Equation 6.4 and 6.5 were also tested with good results. Template algorithms may be rigorously defined in terms of a distance metric [4]. The result from testing is given in Figure 25 and Figure 26 respectively. It should be noted that Equation 6.5 contains a correlation computation which is evident when the squared term is multiplied out. The algorithms in Equations 6.1, 6.4, and 6.5 have been implemented in real time. This report, however, contains only the software for the algorithms in Equations 6.4 and 6.5. The execution times for these algorithms as defined in the equations is on the order of 6.46 ms with a 160 nsec cycle time and a320 nsec memory read cycle.

C. Tracker Description

This section describes the operation of the tracker macrosoftware given in Appendix A. An examination of the software mnemonics show that the code is basically typical minicomputer code as defined in Section 7.A. The exceptions are: WIN, COR, STD, CROSS, FRM, WAT, IMG, REF, MNV, and LAIBUS.

These instructions are covered here briefly and in more detail in Section 7.C. The variables and constants in cells 208 through 418 can be easily accessed in microcode by constants generated in the K bus logic. A complete list of variable names and definitions are given in Figure 27.

The tracker code given in Appendix A contains both adaptive gate centroid code and template matching code. The following sections contain only a description of the template matching code which starts at label START2. The adaptive gate tracker code is documented in a separate report [5].

Tracker operation can be broken into two major computation loops, the scan loop and the track loop. These computation loops are flow charted in Figures 28 and 29 respectively. The hardware reset shown at the top of Figure 28 initializes the tracker hardware and software. When initialization is complete, a crosshair is displayed as an overlay on the TV monitor, centered at column 50, row 50 (NX = NY = 50) as shown in Figure 20. The crosshair is produced by routine CROSS. The operator then aims the camera at the target to be tracked. In roughly .5 seconds the image data is scanned and the GAIN and BIAS latches are set to optimum values. The iterative process of setting the GAIN and BIAS latches continues until the track switch is turned on. At this point the track loop is entered as shown in Figure 29. The AGC conditions existing at the time of track initiation are stored. Then continuing AGC action maintains the quality of the reference in spite of changes in scene brightness and local area responsivity in the CCD TV camera. The reference array is stored

VARIABLE NAME	DEFINITION
SSW7	CONSTANT, USED TO SELECT SSW7
ONEMC1	CONSTANT, USED IN FILTER EQUATION
FRMRQ	CONSTANT
ONE	CONSTANT
IP	CONSTANT, IMAGE POINTER
RP	CONSTANT, REFERENCE POINTER
CP	CONSTANT, CORRELATION POINTER
IP2	(NOT USED)
MEAN	VARIABLE, (MEAN VALUE)*81
STD	VARIABLE, (MEAN DEVIATION)*81
MAXI	VARIABLE, MAXIMUM OF CORRELATION ARRAY
MINI	VARIABLE, MINIMUM OF CORRELATION ARRAY
NX	VARIABLE, CURRENT COLUMN TRACK POINT
NY	VARIABLE, CURRENT ROW TRACK POINT
DELXX	VARIABLE, DELTA ROW TRACK CORRECTION
DELYY	VARIABLE, DELTA COLUMN TRACK CORRECTION
MODE	VARIABLE, CURRENT MODE LATCH
BWC	VARIABLE, CURRENT BLACK/WHITE CONTROL WORD
GAIN	VARIABLE, CURRENT GAIN LATCH VALUE
BIAS	VARIABLE, CURRENT BIAS LATCH VALUE
TP5	CONSTANT, (DEC .5)
TIP5	CONSTANT, (DEC 1.5)
N10	CONSTANT, LIMIT FOR TRACKING AREA
N90	CONSTANT, LIMIT FOR TRACKING AREA
TP5D	CONSTANT, (DEC .5)
TIP5D	CONSTANT, (DEC 1.5)
T7PO	CONSTANT, (DEC 7.0)
T8PO	CONSTANT, (DEC 8.0)
T14P5	CONSTANT, (DEC 14.5)
T13P5	CONSTANT, (DEC 13.5)
T31	CONSTANT, INITIAL VALUE FOR GAIN LATCH
DCBLK	CONSTANT, INITIAL VALUE FOR BIAS LATCH
T81	CONSTANT, (DEC 81)
T81PRM	CONSTANT, (DEC 81)
ONEONE	CONSTANT, BIAS ADJUSTMENT CONSTANT
ONEMC2	CONSTANT, FILTER EQUATION
C2	CONSTANT, FILTER EQUATION
GP933	CONSTANT, GAIN ADJUSTMENT FACTOR
FIFTY	CONSTANT, (DEC 50)
FULLG	CONSTANT, MAXIMUM GAIN LIMIT
FULLB	CONSTANT, MINIMUM BIAS LIMIT
ZERO	CONSTANT
MNSBPK	VARIABLE, BIAS ADJUSTMENT PARAMETER
SDSBPK	VARIABLE, AGC ADJUSTMENT PARAMETER
MNPK	VARIABLE, BIAS ADJUSTMENT PARAMETER
MNSPK	(NOT USED)
SDPK	VARIABLE, AGC ADJUSTMENT PARAMETER
TEMP	VARIABLE, WORKING
MEANO	VARIABLE, OLD VALUE OF MEAN
STDO	VARIABLE, OLD VALUE OF MEAN DEVIATION
C1	CONSTANT, FILTER EQUATION

Figure 27. List of variable names and definitions.

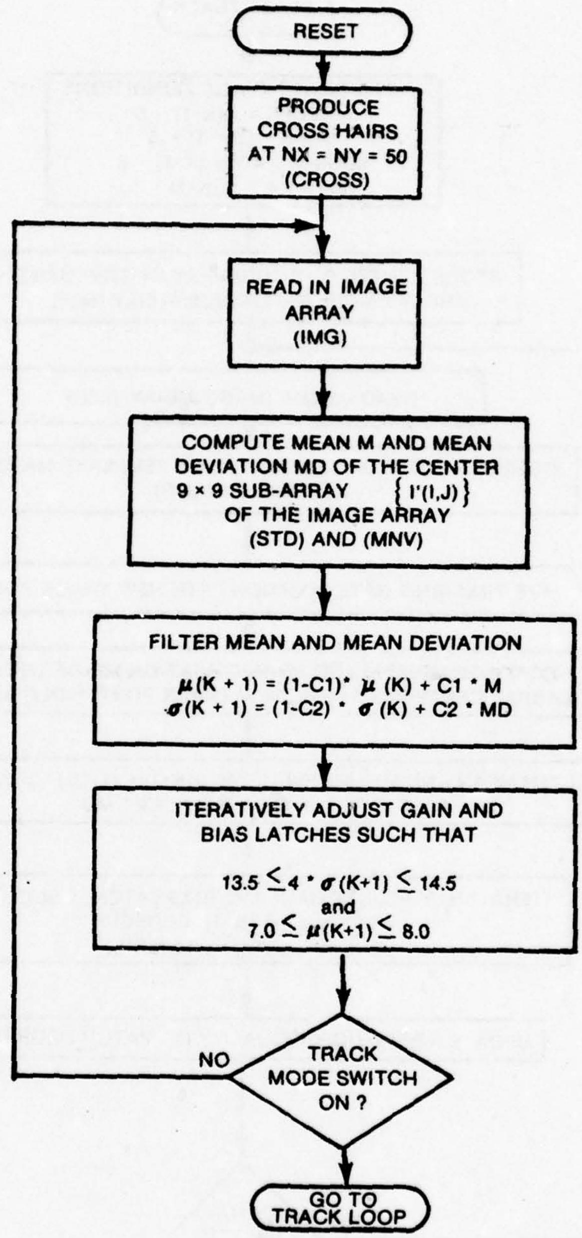


Figure 28. Flow chart of scan loop.

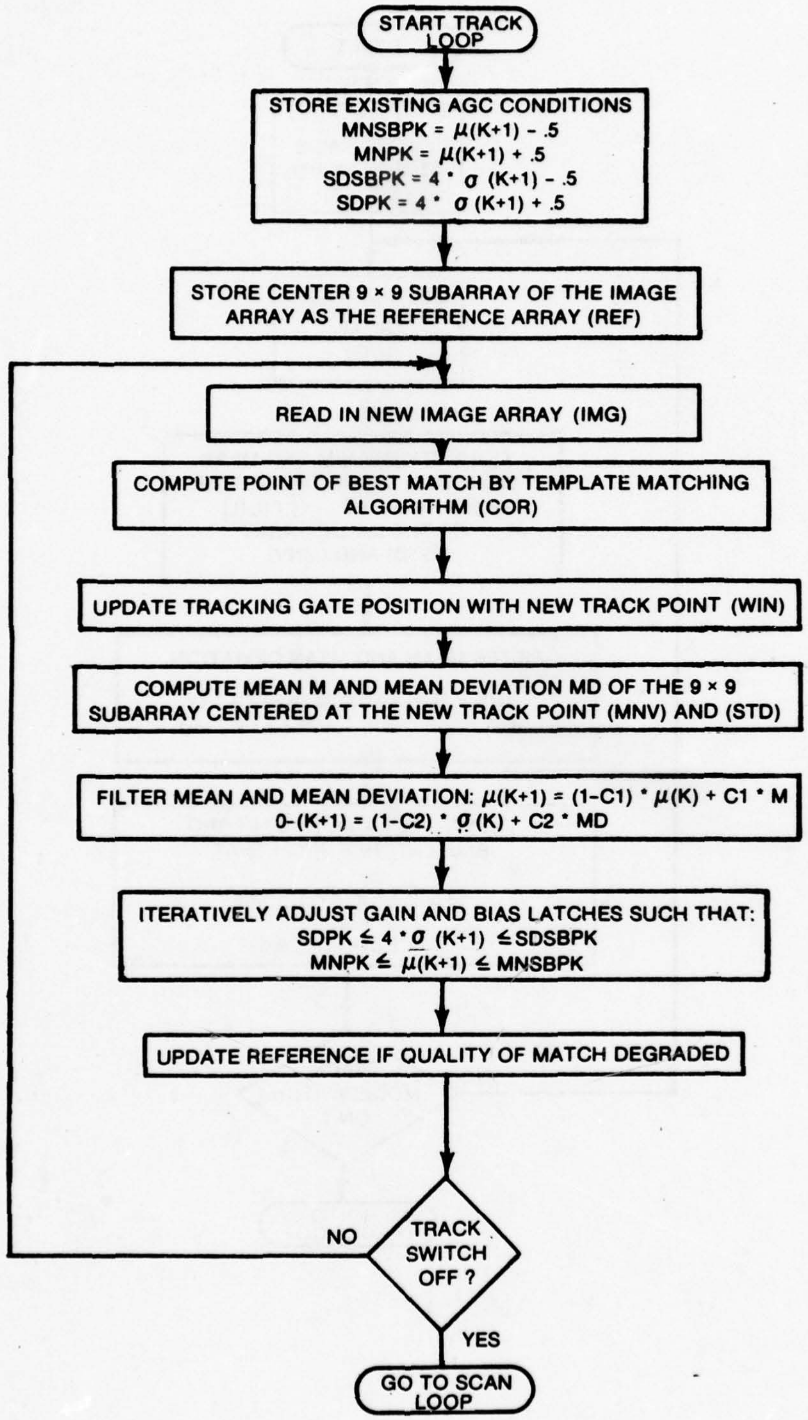


Figure 29. Flow chart of track loop.

and a new image array is read in while computation continues (in parallel). At this point the template matching algorithm finds the point of best match and the tracking gate positions are updated. New mean and mean deviation values are computed for the 9 × 9 subarray centered at the track point. The mean and mean deviation are then filtered and subsequently used by the iterative gain and bias adjustment software. The reference array is updated if the quality of match is degraded. This closes the track loop which starts by reading in a new image array. The iterative gain and bias computations appear in both the scan and track loops. The details of the computation are given in Figure 30. The computation starts after the filtered mean and mean deviation have been computed. The mean is computed by the equation.

$$M = \frac{1}{81} \sum_{I=1}^{9} \sum_{J=1}^{9} I'(I,J)$$

where I′ (I,J) is the 9 × 9 subarray centered at the current track point. Similarly the mean deviation is computed by

$$MD = \frac{1}{81} \sum_{I=1}^{9} \sum_{J=1}^{9} |I'(I,J)-M|$$

The filter equations are given in Figure 28. It should be noted that in the scan loop the following constants are held to fixed values indicated below.

MNPK = 8

MNSBPK = 7

SDPK = 14.5

SDSBPK = 13.5

This choice of the constants has the effect of optimizing the gain and bias settings so that the full range of the analog-to-digital converter is utilized (0 to 15 counts). The effect of successive iterations is that the 9 × 9 subarray has a mean value of 7.5 counts and a dispersion (4σ (K+1)) of 14 counts. Since gain is limited to 5, the dispersion may not reach 14 counts for very low contrast targets, but the mean will always reach 7.5 (filtered value). This technique can cause extremely bright or dark objects to be clipped to 15 or 0 counts respectively, but the overall effect is to enhance the available contrast in the target. The software AGC performed quite well, and enabled the tracker to track the detail that exists in a piece of plain white bond typing paper in preliminary lab tests as shown in Figure 31. Figure 32 shows the enhancement of a very low contrast target which is barely visible on the monitor. Figure 33 shows results of a real time scan and track loop tests on a simulated M48 tank target.

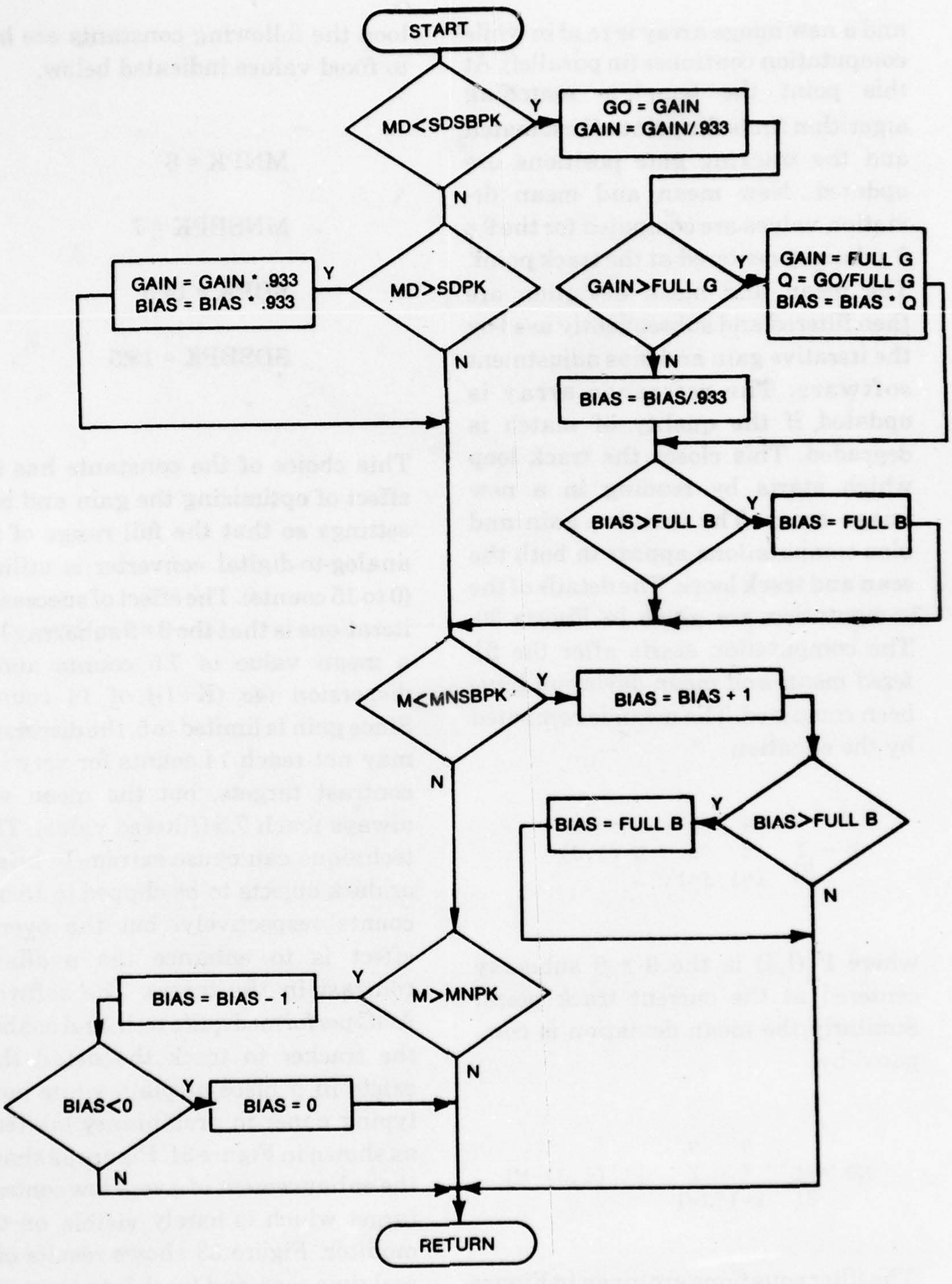
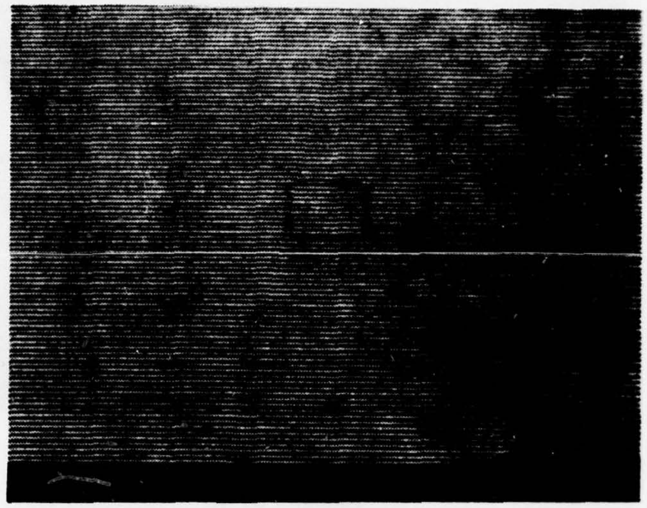
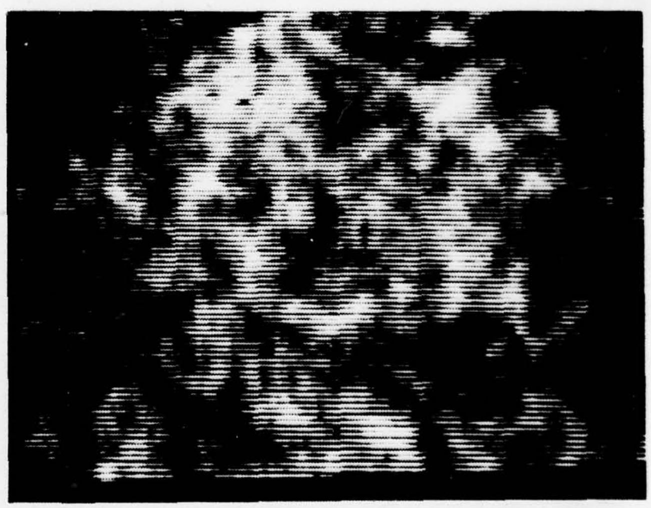


Figure 30. Iterative GAIN and BIAS flow chart.



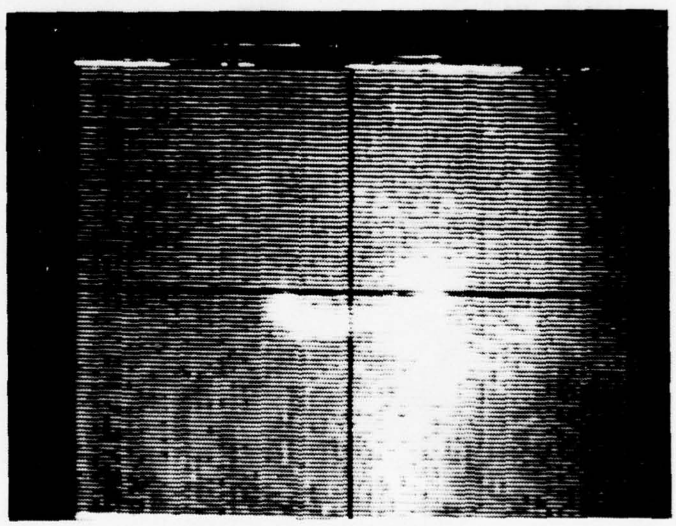
LIVE VIDEO



DIGITIZED AND ENHANCED VIDEO

Figure 31. Plain paper tracking tests.

LOW CONTRAST
CAMERA VIDEO



DIGITIZED AND ENHANCED VIDEO

Figure 32. Results from automatic GAIN and BIAS adjustments.

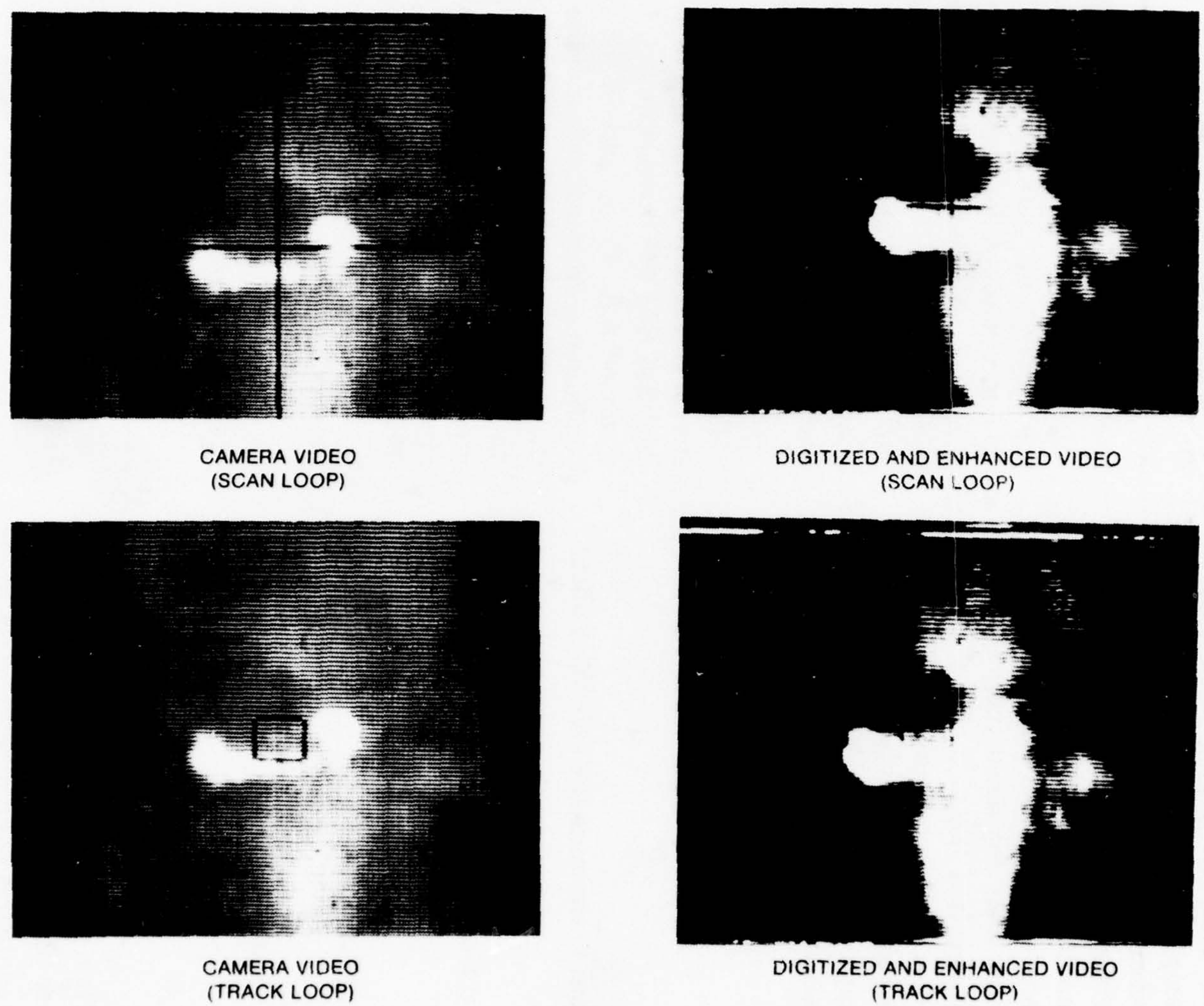


Figure 33. Results of real time scan and track loop tests on a simulated M48 tank target.

The reference update software is not given in this report though several methods for measuring the quality of the template match have been attempted. The methods generally involve the examination of the side lobes of the E(M, N) array about the point of match. No suitable wide range zoom lens was available to make the necessary tests for developing the algorithms. It is hoped that a wide range zoom will be available in FY78 so that the algorithms can be developed. One approach for checking quality is to compute

$$Q = \left\{E(M,N)\right\}\Big|_{max} - \left\{E(M,N)\right\}\Big|_{min}$$

and to examine the side lobes about

$$\{m,n\} = \{(m,n)\,|\,E(m,n)\ \text{minimum}\}$$

Define the set

$$\{D\} = \left\{E(m,n)-E(k,\ell)\,\Big|\,|m-k|=1,\ |n-\ell|=1\right\}$$

Then if $Q<T_q$ and $D|_{min}<T_s$ (T_q and T_s are thresholds) the reference array is updated. A change in the color in the tracking gate has been used as a que to the operator that a reference update has taken place.

7. DETAILED SOFTWARE DESCRIPTION

A. EAI 640 Emulation Code

A subset of the EAI 640 minicomputer instruction set has been emulated on the Intel 3000 microprocessor. A listing of the code is given in Appendix C. The code operates roughly 4 times faster than the EAI 640 and about 2 times faster than the Pacer 100, a newer model EAI digital computer.

The basic set of macroinstructions shown in Figures 34 through 37 is a modified EAI 640 instruction set. The symbols A,X, P, S, and W in the "operation performed" column refer to the microprogrammed machine's registers: accumulator, index register, program counter, stack pointer and status register respectively. The macroinstructions are all 20 bits wide and have two possible formats shown below.

Memory reference

A B	6 bit OP Code	12 bit Displacement

2 bit address option

Other

8 bit OP Code	12 bit Displacement

CODE	BINARY	OCT	
LA	AB00 0000	000	C(E) → A
STA	AB00 0001	001	A → C(E)
LX	AB00 0010	002	C(E) → X
STX	AB00 0011	003	X → C(E)
A	AB01 0100	004	A+C(E) → A
S	AB00 0101	005	A-C(E) → A
M	AB00 0110	006	A*C(E) → A
D	AB00 0111	007	A/C(E) → A
AOM	AB00 1000	010	A+1 → A
OR	AB00 1001	011	AVC(E) → A
XOR	AB00 1010	012	A+C(E) → A
AND	AB00 1011	013	A C(E) → A
C	AB00 1100	014	SET FLAG FOR SKIP INSTRUCTIONS
NOT USED	AB00 1101	015	
NOT USED	AB00 1110	016	
NOT USED	AB00 1111	017	
J	AB01 0000	020	D → P (OR C(E) → P)
L	AB10 0000	040	P → S, S+1 → S, D → P (OR C(E) → P)

	ADDRESS OPTION	OP CODE	DISPLACEMENT
MEMORY REFERENCE CODE	Z	OP	D
	2 bits	6 bits	12 bits

AB	ADDRESS OPTION
00	Direct
01	Indirect
10	Indexed
11	Indirect, Indexed

Figure 34. Memory reference instructions.

CODE	BINARY	OCT	
CLR	0011 0000	060	0 → A
AOA	0011 0001	061	A + 1 → A
CAO	0011 0010	062	1 → A
TCA	0011 0011	063	1 + A → A
ARS	0011 0100	064	ARITHMETIC RIGHT SHIFT (D TIMES)
ALS	0011 0101	065	ARITHMETIC LEFT SHIFT (D TIMES)
LRS	0011 0110	066	LOGICAL RIGHT SHIFT (D TIMES)
SSP	0011 0111	067	SET SIGN BIT OF A REG TO ZERO
SSN	0011 1000	070	SET SIGN BIT OF A REG TO ONE
EX	0011 1001	071	X → A and A → X
EP	0011 1010	072	A → P and P → A
ES	0011 1011	073	A → W and W → A
ICX	0011 1100	074	X + D → X, SKIP NEXT IF SIGN BIT CHANGES
DCX	0011 1101	075	X - D → X, SKIP NEXT IF SIGN BIT CHANGES

INSTRUCTION FORMAT FOR REGISTER OPERATIONS $\overbrace{\underbrace{\text{OP}}_{\text{8 bits}}}^{\text{OP CODE}}$ $\overbrace{\underbrace{\text{D}}_{\text{12 bits}}}^{\text{DISPLACEMENT}}$

Figure 35. Register operation instructions.

CODE	BINARY	OCT	
SE	0111 0000	160	SKIP IF C(E) = A
SG	0111 0001	161	SKIP IF A > C(E)
SL	0111 0010	162	SKIP IF A < C(E)
SNE	0111 0011	163	SKIP IF A ≠ C(E)
SGE	0111 0100	164	SKIP IF A ≥ C(E)
SLE	0111 0101	165	SKIP IF A ≤ C(E)
SKN	0111 1010	172	SKIP IF A < 0
SKP	0111 1011	173	SKIP IF A > 0
SO	0111 1100	174	SKIP IF PREVIOUS CAUSED OVERFLOW
SNO	0111 1101	175	SKIP IF PREVIOUS CAUSED NO OVERFLOW
SAE	0111 1110	176	SKIP IF A REG EVEN
NOT USED	0111 1111	177	—
ICX	0011 1100	074	SEE REGISTER OPERATIONS
DCX	0011 1101	075	SEE REGISTER OPERATIONS

INSTRUCTION FORMAT FOR SKIP INSTRUCTIONS

OP	D
8 bits	12 bits

*Valid following compare instruction (C) only.

Figure 36. Skip instructions.

CODE	BINARY	OCT	
NOP	0011 1110	076	NO OPERATION
P	0011 1111	077	PAUSE
PUSA	0111 0110	166	PUSH ALL REGISTERS ON STACK, S + 12 → S
PUSX	0111 0111	167	PUSH A, X, P, W ON STACK, S + 4 → S
POPX	0111 1000	170	POP A, X, P, W FROM STACK, S - 4 → S
RTN	0111 1001	171	POP P FROM STACK, S - 1 → S
DI	1111 1100	374	DEVICE D → A
DO	1111 1101	375	A → DEVICE D
NOT USED	1111 1110	376	
NOT USED	1111 1111	377	

OTHER INSTRUCTION FORMAT→(SAME AS REGISTER OPERATIONS)

Figure 37. Other Instructions.

In the case of memory reference instructions, there are four addressing options.

	AB
Direct	00
Indirect	01
Indexed	10
Indirect, Indexed	11

In Figure 34, the symbol E is used to indicate an effective address formed in one of the four possible ways of. If D is the symbol for a 12 bit displacement in the instruction word, then

Direct E = D
Indirect E = C(D)

Indexed E = X + D (sum of Index Register and D)

Indirect, Indexed E = C(D) + X

where the symbol C(D) represents the content of memory location D.

The register assignments in the Intel 3002 chip are made as follows:

Microprogrammed Machine	Intel 3002 Register
A	R0
X	R1
	R2 (Working)
P	R3
S	R4
	R5 (Working)
	R6 (Working)
W	R7
	R8 (Working)
	R9 (Working)
	AC (Working)
	T (Working)

Registers R2, R5, R6, R8, R9, AC, and T are used as working registers in microcode operations. Registers R0, R1, R3, R4, and R7 are maintained for the microprogrammed maching. The A and X registers are used as conventional accumulator and index registers respectively. The P register is the program counter and contains the address of the next instruction to be executed. The S register is the stack pointer and contains the next available address for the storage stack located in main memory. The W register is used as a status indicator and will be used in interrupt code.

When the front panel reset switch is activated, a logic high is placed on pin 36 of the Intel 3001 microprogram control unit. The logic high forces the instruction at location 0008 in micromemory to be executed. The initialization code for the microprocessor is placed at this location.

The last instruction in the initialization code is a JZR (FETCH). FETCH is a label for row zero column

15 and is the starting location for the instruction fetch code. The initialization code is given below and assumes that the initial value of the stack pointer and program counter are located in cells 0 and 1 of main memory respectively.

The instruction fetch code is located in row zero column 15 (OOF16), and is used to fetch the next instruction to be executed from macromemory. The code includes a mask so only the 12 bit displacement is loaded into the AC register of the Intel 3002. The AC

```
000H:      INIT:      CLR (A);                          0→A
                      CLR (X);                          0→X
                      CLR (W);                          0→W
                      CLR (T);                          0→T
                      LMI (T);                          T→MAR
/*WRITE ZERO TO INTERRUPT STRUCTURE*/
                      ILR (W)        ROT;               W AC
                      LMI (T)        FF1 RRM;           0 MAR, T + T
                      ACM (AC);                         C (0) Ac AC
/* C ()) = INITIAL VALUE OF STACK POINTER*/
                      SDR (S)        FF1;               AC→S
                      LMI (T)        ff1 RRM;           1→MAR, T + 1→T
                      ACM (AC);                           C (1)→AC
/* C (1) =STARTING ADDRESS OF PROGRAM*/
                      SDR (P) FF1                       JZR (FETCH)
```

register is also copied into R9 for later usage. The last instruction in the fetch sequence contains a JPX instruction which causes a 16-way branch to the various "classes" of instructions. Figure 38 shows all conditional branches used to determine the macroinstruction to be executed. Note that many unused instructions are shown in Figure 38 (FETCH), allowing for future growth to more complex macroinstructions. The fetch code also

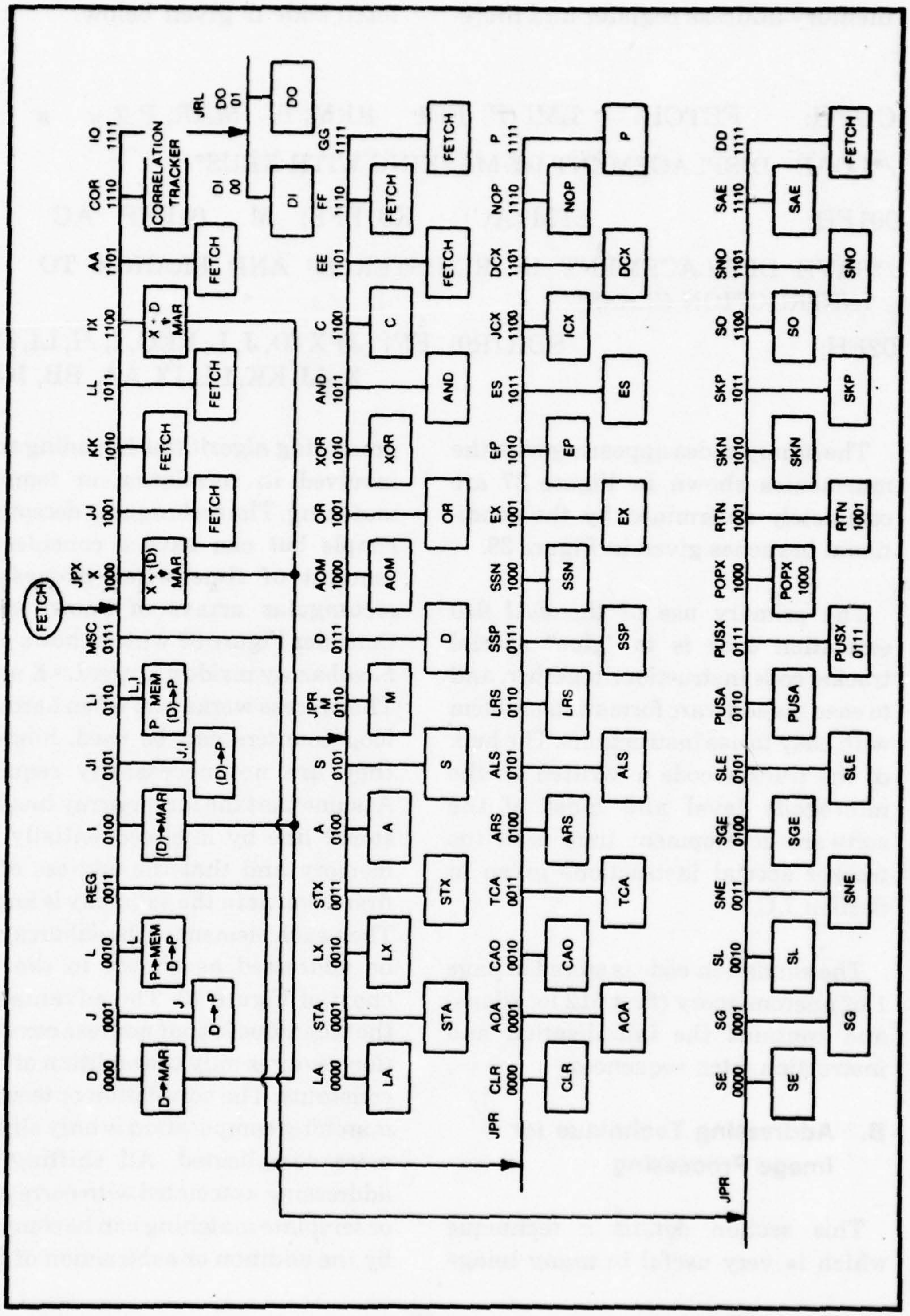


Figure 38. Conditional branches used to select microcode sequence.

forces the P register contents to the memory address register and increments the P counter by one. The fetch code is given below.

```
OOFH:    FETCH:    LMI (P) FF1    RRM; P    MAR, P Z π    π
/*LOAD  DISPLACEMENT BY MASKING WITH KBUS*/
001FH;             LTM (AC)    K01FFF;  M    01FFP  AC
/*SAVE  DISPLACEMENT  IN  REGISTER  R9  AND  BRANCH  TO
  INSTRUCTION CLASS*/
02FH;              SDR (R9)  FF1  JPX (D, J, L, REG, I, JI, LI, MSC,
                                   X, JJ, KK, LL, IX, AA, BB, IO);
```

The binary codes appearing with the mnemonics shown in Figure 37 are completely determined by the conditional branches given in Figure 38.

The primary use of the EAI 640 emulation code is to "glue" special tracker code instructions together, and to ease the software formation problem with easy to use instructions. The bulk of the tracker code is written on the microcode level and most of the software development time is in the tracker special instructions given in Section 7.C.

The emulation code is stored in page 1 of micromemory (first 512 locations) and contains the initialization and instruction fetch sequences.

B. Addressing Technique for Image Processing

This section details a technique which is very useful in many image processing algorithms including those involved in correlation or template matching. The technique is deceptively simple but can save a considerable amount of time when processing rectangular arrays of known sizes. Consider Figure 39 which shows a N × M subarray inside a larger L × K array. The process works best when hardware loop counters can be used, however, they are not necessarily required. Assume that the larger array has been stored line by line, sequentially into memory and that the address of the first element in the subarray is known. Then each element of the subarray can be addressed as shown in the flow chart of Figure 40. The advantage of the technique is that address computation requires only the addition of fixed constants. The correlation or template matching computation is only slightly more complicated. All shifting and addressing associated with correlation or template matching can be computed by the addition or subtraction of fixed

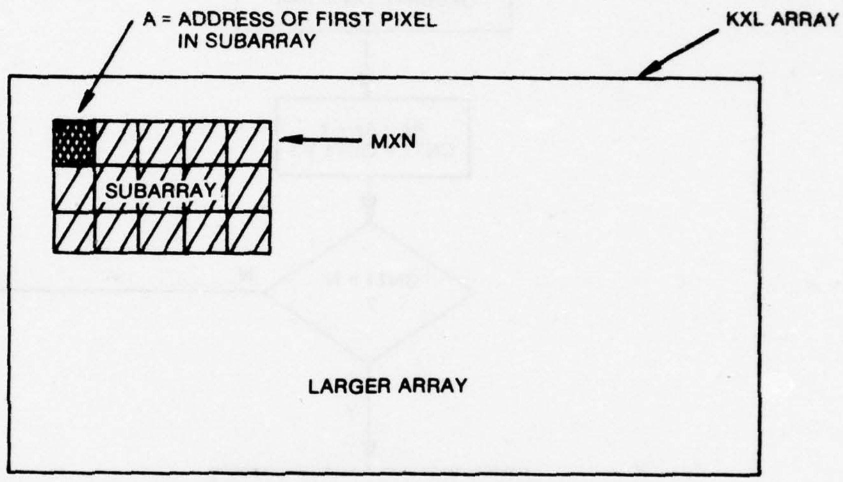


Figure 39. Subarray inside larger array.

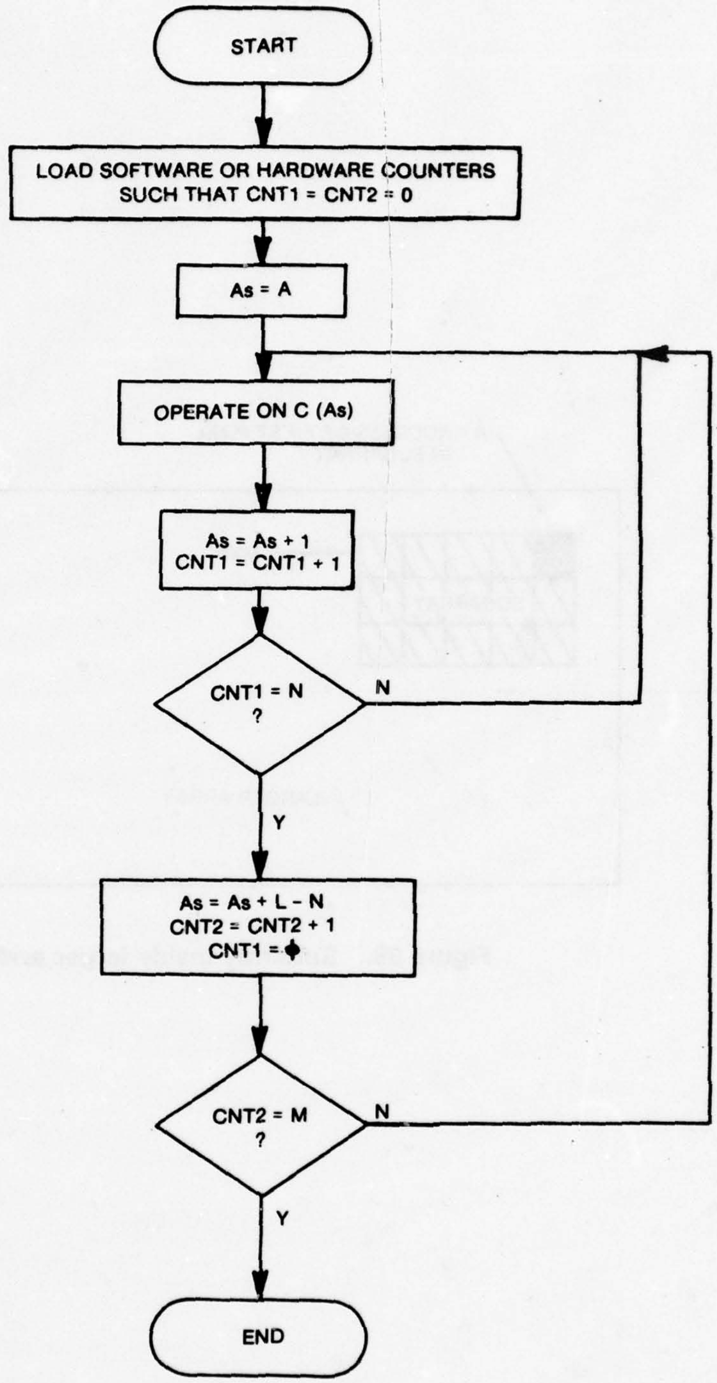


Figure 40. Addressing technique for image processing.

constants. Multiplications for address formation are eliminated. An example of template matching addressing can be found in the flow chart of Figure 46.

C. Tracker Microcode

(1) Routine WIN. Routine WIN is used to generate tracking gates to indicate track point on the TV monitor. The track point is defined by (NX, NY) the column and row locations of the tracking gate center. Figure 19 shows the 9 × 9 array which is inclosed by the tracking gate of fixed size, and Section 3.C gives a description of the latches which must be loaded to generate a tracking gate on the TV monitor.

The variables NX and NY are stored in macromemory, and routine WIN reads these values along with the black/white control word which defines the color of the tracking gate.

The mode latch is forced to the tracking gate mode with the requested tracking gate color. The mode latch status word is also updated to give the current mode latch status. Figure 41 is a flow chart of WIN and shows the addressing and loading of the tracking gate control latches with the 3002 MAR register.

(2) Routine CROSS. Routine CROSS is similar to WIN except that the mode latch is enabled to draw a crosshair. The right and left column latches are loaded with NX and the upper and lower latches are loaded with NY. The crosshair color, as in routine WIN, is determined by the black/white control word in macromemory.

The flow chart for CROSS is shown in Figure 42.

(3) Routine FRM. Routine FRM is analogous to a shutter release on a camera. Execution of FRM causes a new frame to be loaded into image memory (10K memory).

The flow chart for FRM, shown in Figure 43, contains all operations performed in the routine.

(4) Routine WAT. Routine WAT is used only after FRM has been executed. Routine WAT tests the sign bit of the I bus to determine when a frame has been completely stored. If the frame is not ready, a wait loop is entered until the sign bit goes high (frame ready). The flow chart for WAT is given in Figure 44.

(5) Routine IMG. Routine IMG is used to read the image array I(I,J), as shown in Figure 19, into macromemory stored starting at address IP (image pointer). The image pointer IP address is located at 20 octal in macromemory. The image is stored line by line, sequentially in memory. The routine is detailed in the flow chart shown in Figure 45. Since image data is read into macromemory during the

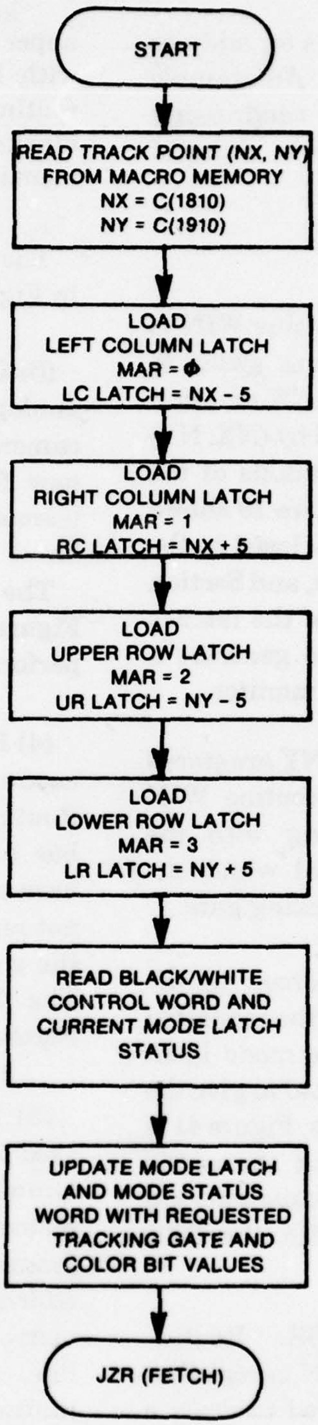


Figure 41. Flow chart for routine WIN.

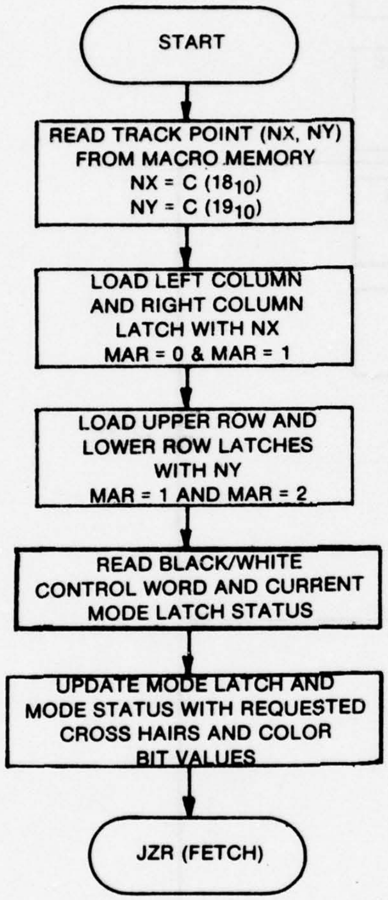


Figure 42. Flow chart for routine CROSS.

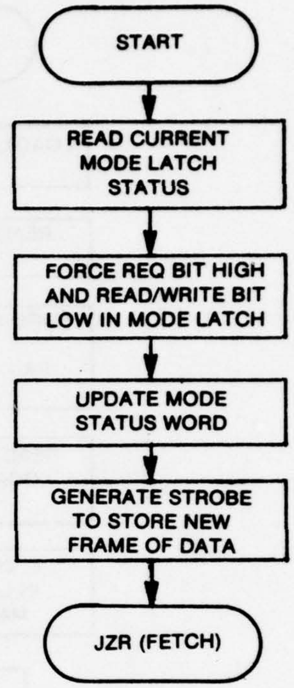


Figure 43. Flow chart for routine FRM.

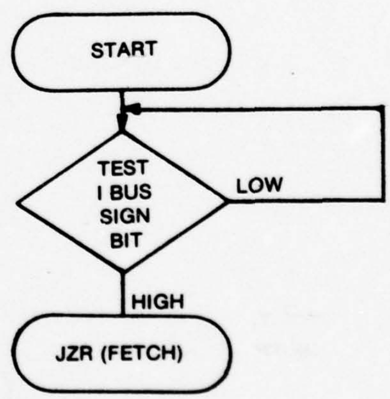


Figure 44. Flow chart for routine WAT.

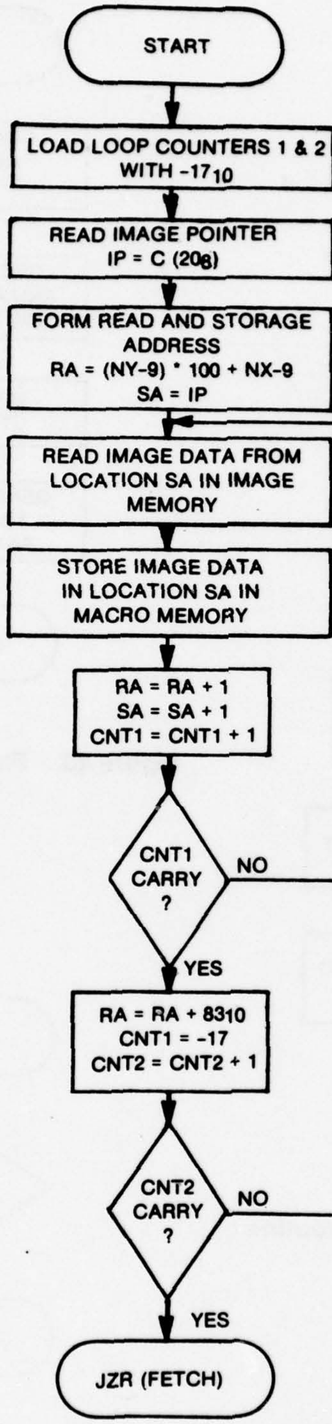


Figure 45. Flow chart for routine IMG.

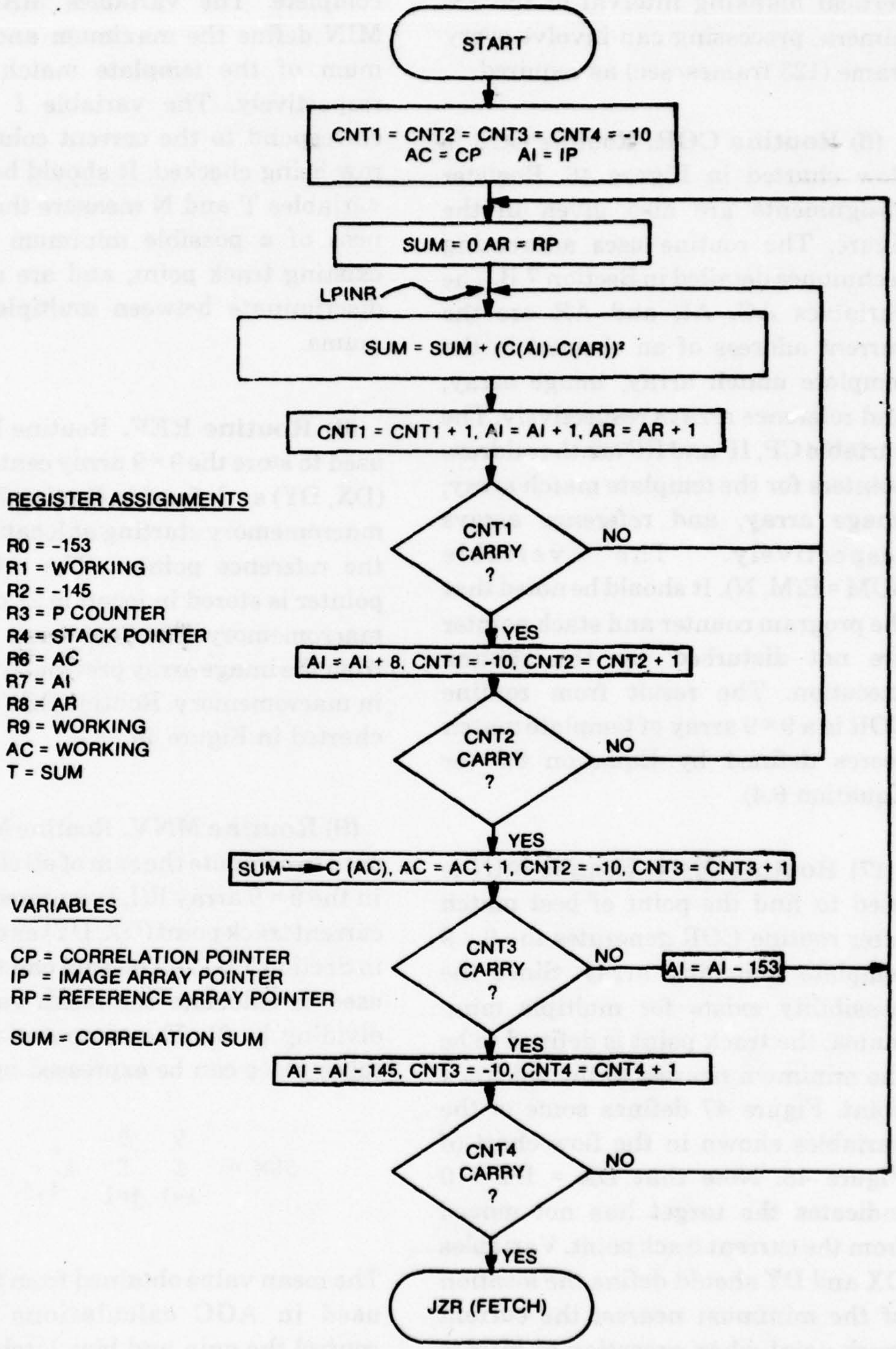


Figure 46. Flow chart for routine COR.

vertical blanking interval of the TV camera, processing can involve every frame (123 frames/sec) as required.

(6) Routine COR. Routine COR is flow charted in Figure 46. Register assignments are also given in the figure. The routine uses addressing techniques detailed in Section 7.B. The variables AC, AI, and AR are the current address of an element in the template match array, image array, and reference arrays respectively. The variable CP, IP and RP are the address pointers for the template match array, image array, and reference arrays respectively. The variable SUM = E(M, N). It should be noted that the program counter and stack pointer are not disturbed by the routine execution. The result from routine COR is a 9 × 9 array of template match scores defined by Equation 6.5 (or Equation 6.4).

(7) Routine MIN. Routine MIN is used to find the point of best match after routine COR generates the 9 × 9 template matching array. Since the possibility exists for multiple minimums, the track point is defined to be the minimum nearest to the old track point. Figure 47 defines some of the variables shown in the flow chart of Figure 48. Note that DX = DY = 0 indicates the target has not moved from the current track point. Variables DX and DY should define the location of the minimum nearest the current track point when execution of MIN is complete. The variables MAX and MIN define the maximum and minimum of the template match array respectively. The variable I and J correspond to the current column on row being checked. It should be noted variables T and N measure the closeness of a possible minimum to the existing track point, and are used to discriminate between multiple minimums.

(8) Routine REF. Routine REF is used to store the 9 × 9 array centered at (DX, DY) as defined in Section 7.C(7) in macromemory starting at location RP, the reference pointer. The reference pointer is stored in location 21 octal in macromemory. The data array is read from the image array previously stored in macromemory. Routine REF is flow charted in Figure 49.

(9) Routine MNV. Routine MNV is used to compute the sum of all elements in the 9 × 9 array R(I,J) centered in the current track point (DX, DY) as defined in Section 7.C.(7). The sum can then be used to calculate the mean value by dividing by 81. The computation that takes place can be expressed by

$$SUM = \sum_{i=1}^{9} \sum_{j=1}^{9} R_{i,j}$$

The mean value obtained from SUM is used in AGC calculations which control the gain and bias latches.

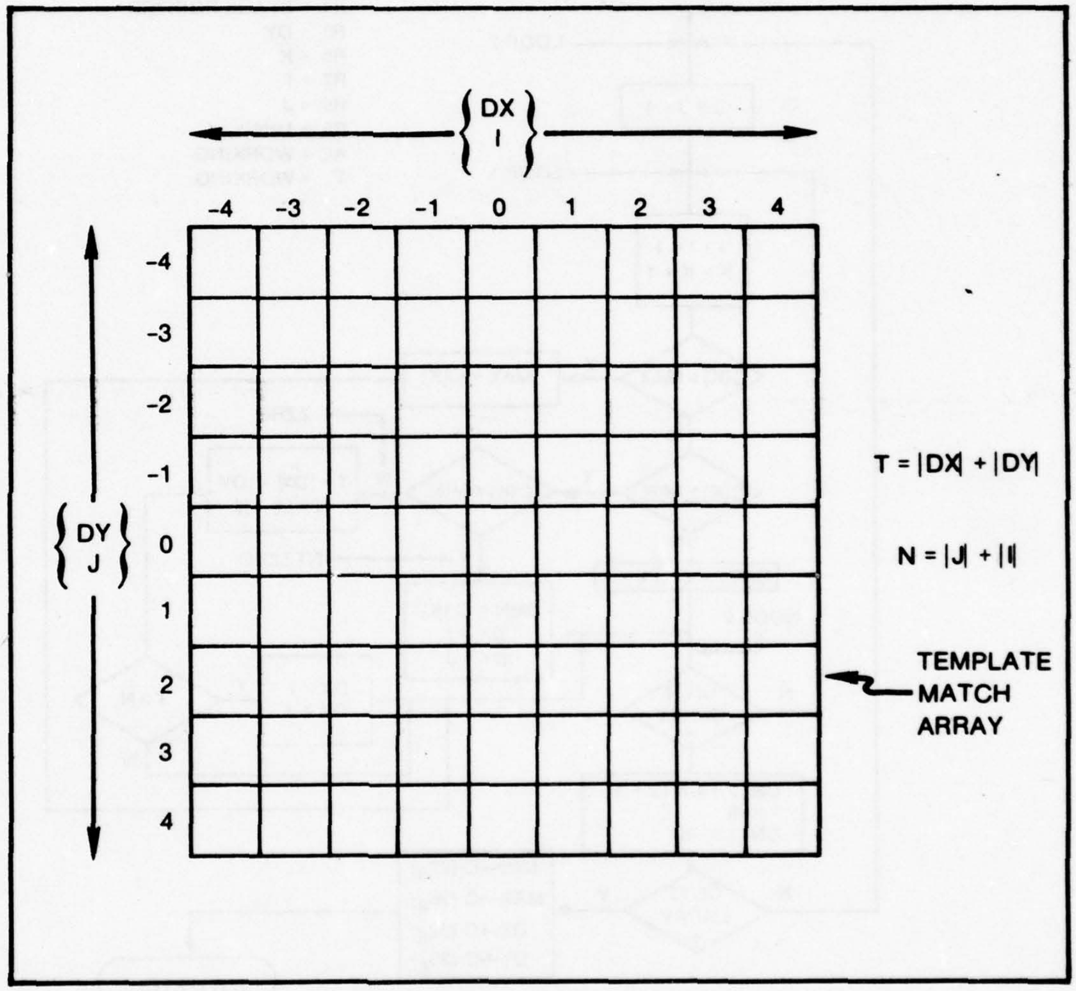


Figure 47. Variable definition for routine MIN.

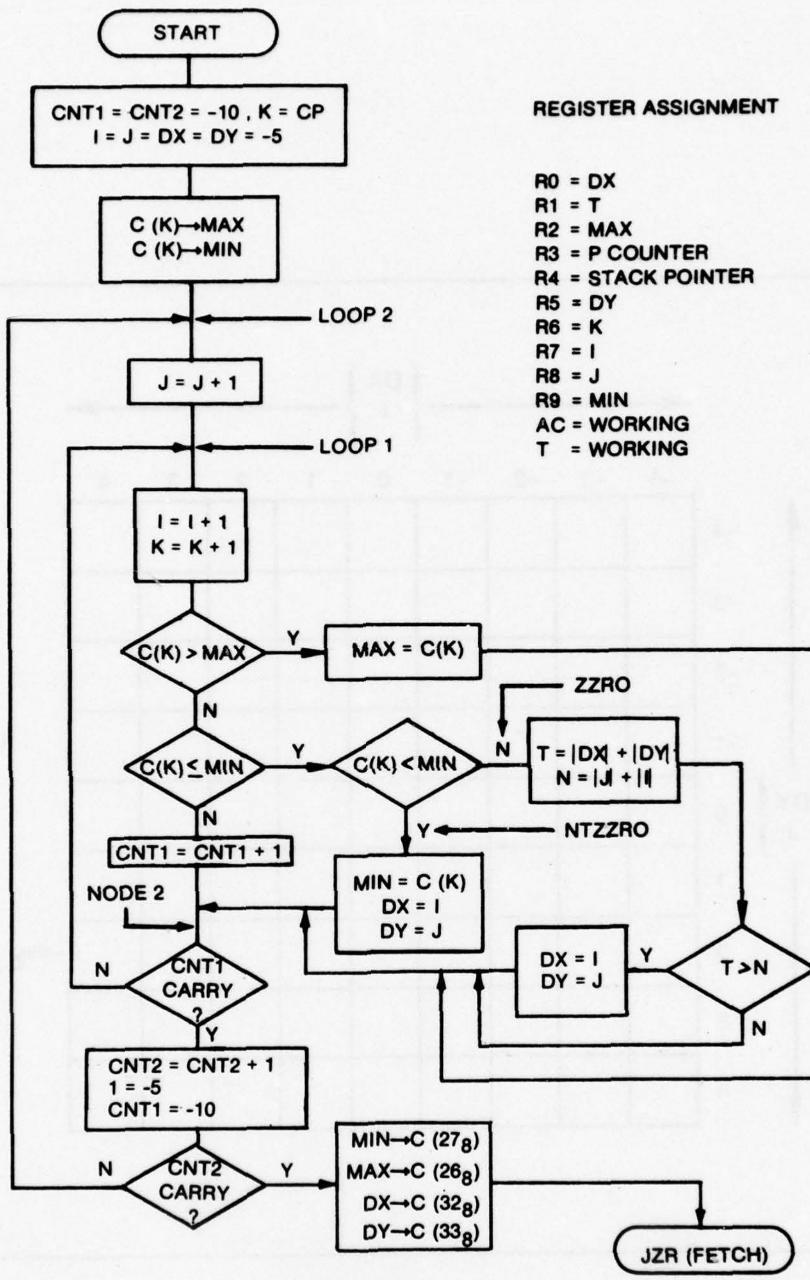


Figure 48. Flow chart for routine MIN.

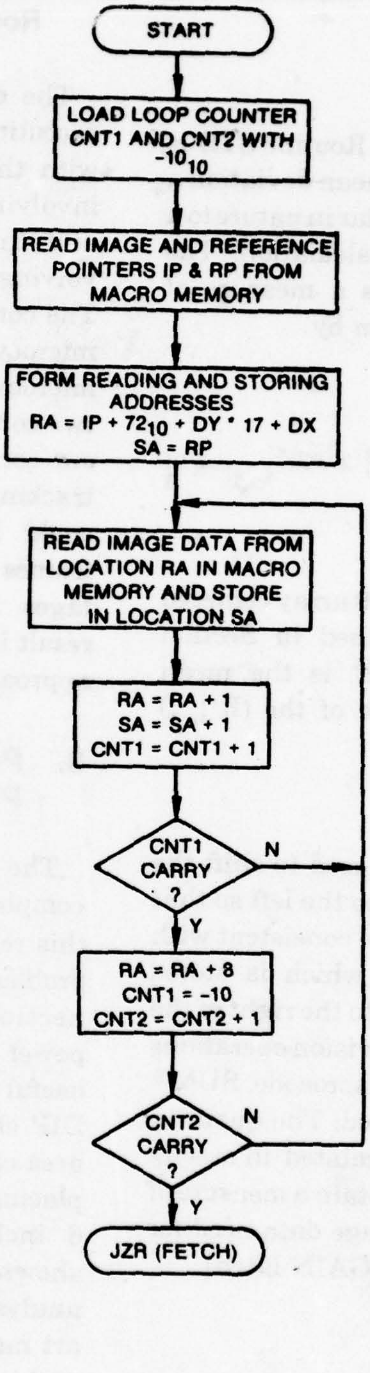


Figure 49. Flow chart for routine REF.

The flow chart for routine MNV is shown in Figure 50.

(10) Routine STD. Routine STD is used to calculate the mean deviation a quantity which is similar in nature to a standard deviation calculation. The quantity calculated is a measure of dispersion and is given by

$$SUM = 81 * \sigma_A = 4 \sum_{i=1}^{9} \sum_{j=1}^{9} | 2^5 * R'_{i,j} - \overline{R}' |$$

where ($R'_{i,J}$) is the 9 × 9 array centered at (DX, DY) as defined in Section 7.C.(7), and where $\overline{R}'$ is the mean value of the elements of the ($R'_{i,J}$) array.

The constant 2^5 is used to shift the image data 5 places to the left so that the image data will be consistent with the mean value $\overline{R}'$ which is stored with 5 binary places to the right of the binary point. Since division operations must take place in macrocode, SUM = 81 * σA is calculated. The quantity σA = SUM/81 is calculated in macrocode and is used to obtain a measure of the dispersion of image data need for gain determination (GAIN latch).

Figure 51 shows a flow chart of the calculations performed in STD.

D. Execution Times for Tracker Routines

The chart in Figure 52 shows the execution time of each routine along with the number of microcycles not involving a read or write (μ cycles), and the number of microcycles involving a read or write (R/W μ cycles). The computations assume a 160 nsec microcycle and a 320 nsec read or write microcycle. It is of interest to note that two correlation tracking operations or one correlation and one adaptive gate tracking operation (simultaneously) could be accomplished at over 60 frames a second. Though the advantages are not clearly defined, this result is indicative of the power of the approach.

8. POWER CONSUMPTION PACKAGING

The packaging problem is a very complex one for the system given in this report, and it is clear that major problems will stem from interconnections of components and high power consumption. However, it is useful to consider chip area used by the DIP chips in the tracker. The needed area can be graphically illustrated by placing all tracker components on two 6 inch diameter circular areas as shown in Figures 53 and 54. The area analysis assumes the use of state of the art memory components available as of this writing and includes only those chips needed for tracker operation. No

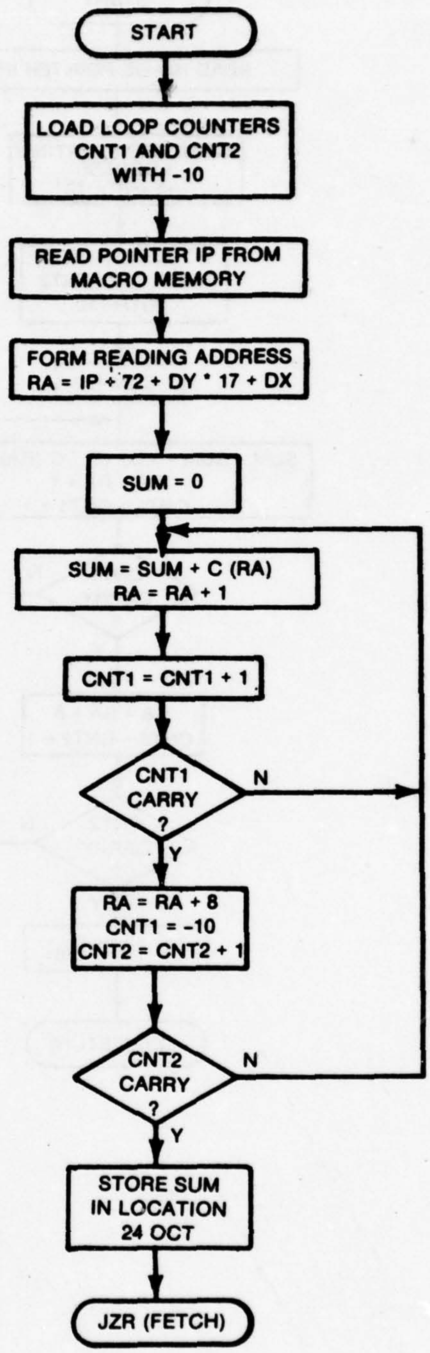


Figure 50. Flow chart for routine MNV.

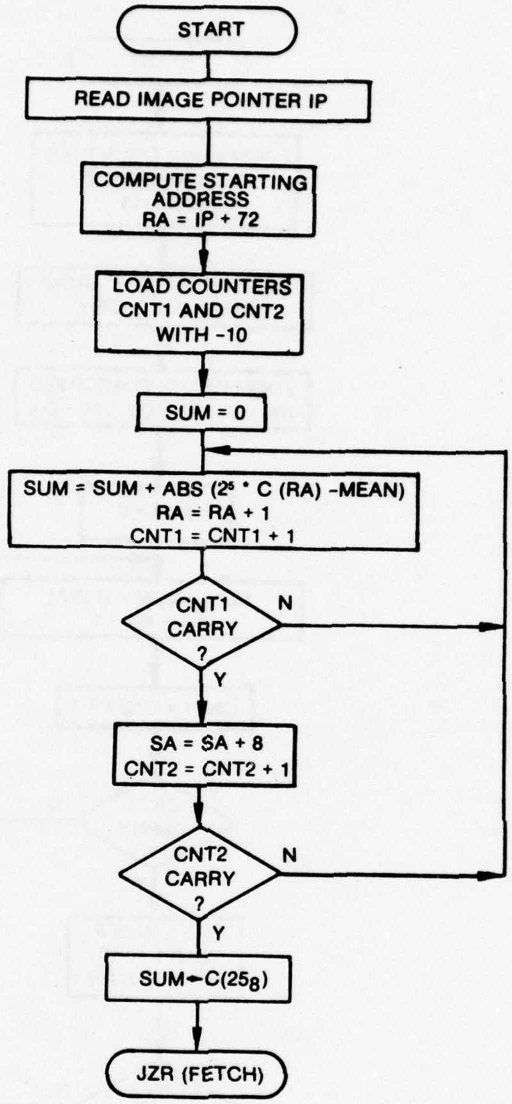


Figure 51. Flow chart for routine STD.

ROUTINE	CYCLE LESS R/W	READS (R)	WRITES (W)	EXECUTION TIME (m sec)
WIN	4	6	36	.0089
CROSS	4	6	21	.0066
FRM	1	2	13	.0030
IMG	85	86	291	.1013
COR	13,127	81	14,137	6.4884
MIN	245	324	3,399	.7259
REF	85	81	112	.0726
MNV	84	1	119	.0462
STD	85	1	1,168	.2144
WAT	0	0	5	.0000*
19,301	13,720	588	TOTAL	7.67

Figure 52. Estimated execution times of tracker routines.

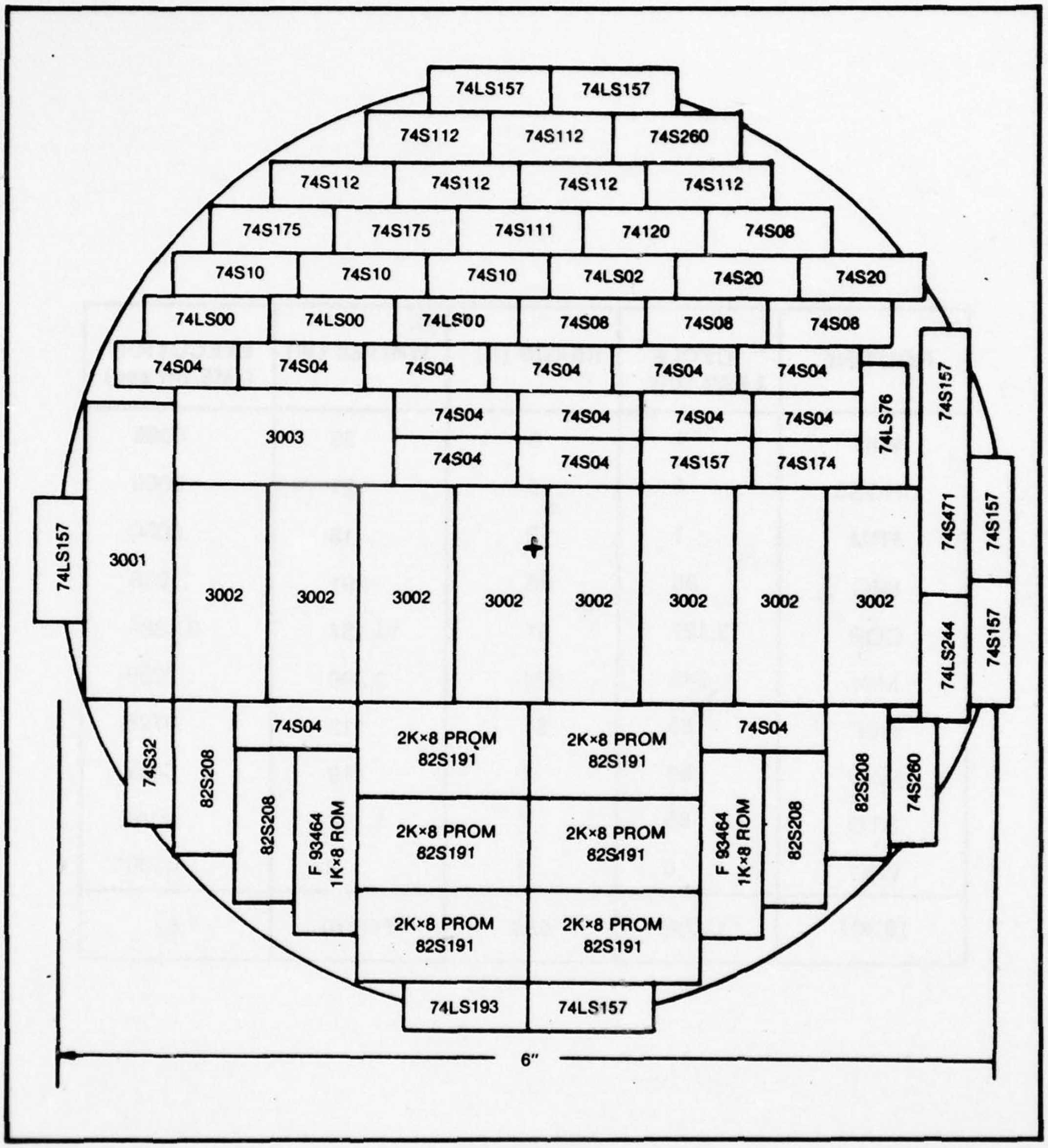


Figure 53. Graphic illustration of DIP area needed for tracker.

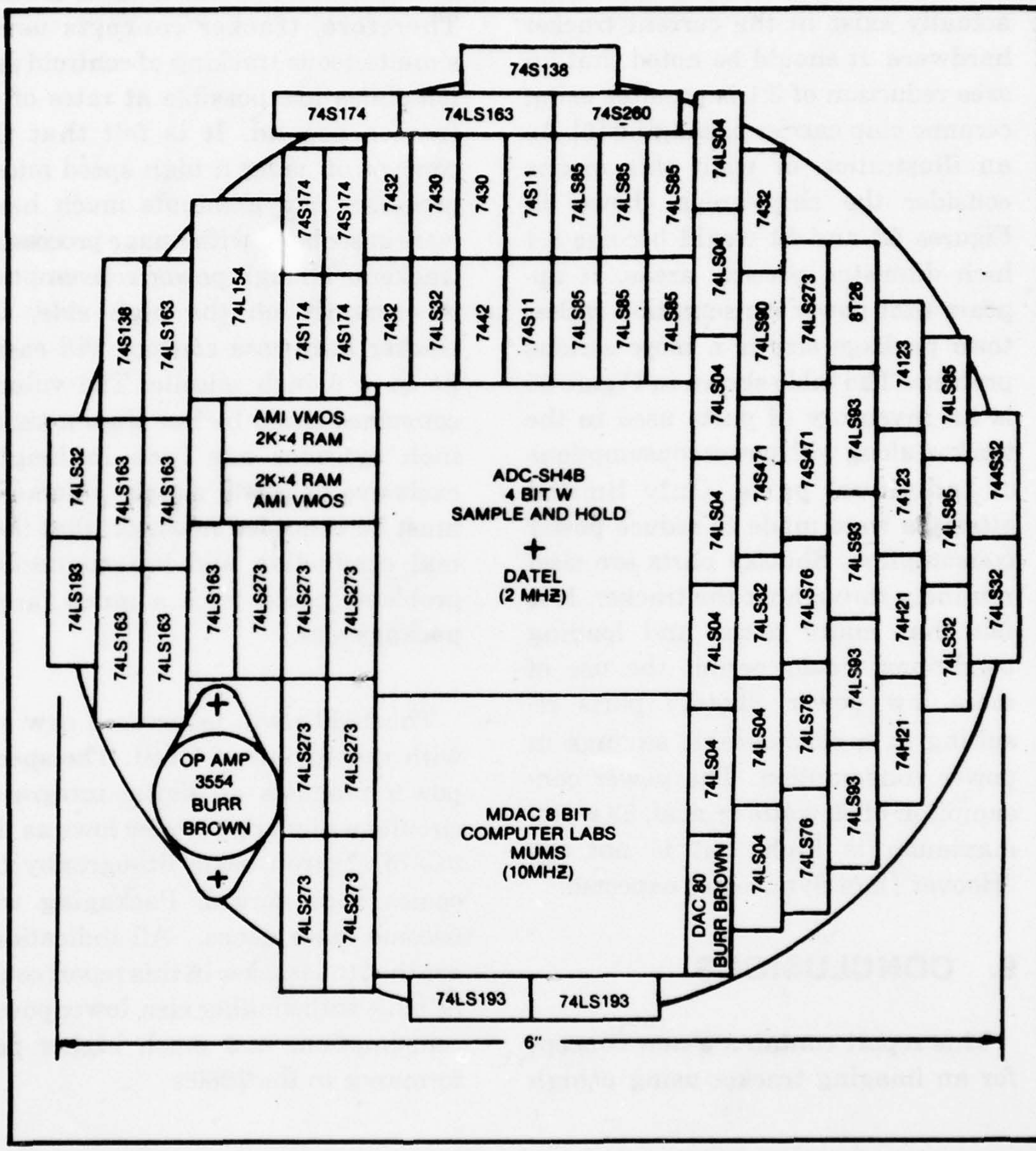


Figure 54. Graphic illustration of DIP chip area needed for tracker.

special attempts were made to reduce area, except in the case of memory, analog to digital conversion, video control latches, and operational amplifiers. The bulk of the chips shown actually exist in the current tracker hardware. It should be noted that an area reduction of 3:1 is possible using ceramic chip carrier packaging [6]. As an illustration of what this means consider the chip areas shown in Figures 53 and 54 would become 3.4 inch diameter circular areas. It appears that power consumption rather than package size is a more serious problem. The table shown in Figure 55 is an inventory of parts used in the tracker, along with power consumptions of individual parts. Only limited attempts were made to reduce power consumption. Shottky parts are used routinely throughout the tracker. It is felt that many paths and loading conditions would permit the use of more low power Shottky parts resulting in a considerable savings in power consumption. The power consumption of 39 watts typical, 56 watts maximum is high, but is not the "Hoover Dam Syndrome" expected.

9. CONCLUSIONS

This report contains a new concept for an imaging tracker using a high speed microprocessor. The tracker, capable of adaptive gate centroid tracking and template matching tracking, achieved each type of tracking at frame rates up to 123 frames/second. Therefore, tracker concepts using simultaneous tracking of centroid and templates are possible at rates of 60 frames/second. It is felt that the concept of using a high speed microprocessor can eliminate much hardware associated with image processing trackers. Though power consumption is currently on the high side, the tracker hardware concept will easily fit in a 6 inch missile. The volume consumed could be less than a six (6) inch cylinder one inch in height, exclusive of power supply volume. It must be conceded however, that thermal conduction and interconnection problems could force a much larger package size.

The 1980's will usher in a new era with the advent of VLSI. The speed-power products of digital integrated circuits will plunge to new lows as the use of electron beam lithography becomes wide spread. Packaging will become more dense. All indications are that the tracker in this report could be built with smaller size, lower power consumption, and much higher performance in the 1980's.

NO CHIPS	PART	FUNCTION AND COMMENTS	TYP	MAX	NO PINS
8	3002	CENTRAL PROCESSING ELEMENT	5.800	7.600	28
1	3003	LOOK-AHEAD CARRY GENERATOR	.400	.650	28
1	3001	MICROPROGRAM CONTROL UNIT	.850	1.200	40
22	74S04	HEX INVERTER	3.3	5.940	14
5	74S174	HEX D FLIP-FLOP WITH CLEAR	2.250	3.600	16
1	74S11	DUAL JK FLIP-FLOP	.07	.1025	14
4	74S157	QUAD 2 TO 1 MULTIPLEXER	1.0	1.560	16
1	74LS244	OCTAL BUFFER WITH 3-STATE OUTPUTS	.135	.230	20
3	74LS00	POSITIVE NAND GATE	.036	.066	14
4	74S08	POSITIVE NAND GATE	.640	1.140	14
3	74S10	POSITIVE NAND GATE	.225	.405	14
1	74LS02	POSITIVE NOR GATE	.007	.013	14
2	74S20	POSITIVE NAND GATE	.100	.180	14
2	74S175	QUAD D FLIP-FLOP	.600	.960	16
1	74120	DUAL PULSE SYNCRONIZER	.255	.450	16
6	74S112	DUAL JK FLIP-FLOP, CLEAR AND PRESENT	.450	.750	16
3	74S260	POSITIVE NOR GATE	.390	.675	14
5	74LS32	POSITIVE OR GATE	.122	.245	14
4	74LS76	DUAL JK FLIP-FLOP, CLEAR AND PRESET	.080	.160	16
1	74LS90	DECADE COUNTER	.045	.075	14
4	74LS93	BINARY COUNTER	.180	.780	14
1	8T26	TRANSCEIVER	.435	.435	16
2	74H21	POSITIVE NAND GATE	.200	.320	14
2	74123	RETRIGGERABLE MONSTABLE	.460	.660	16
8	74LS85	4 bit MAGNITUDE COMPARITOR	.416	.800	16
2	74S11	POSITIVE AND GATE	.240	.420	14
7	74LS273	OCTAL D FLIP-FLOP WITH CLEAR	.595	.945	20
1	7442	4 TO 10 LINE DECODER	.140	.280	16
2	7430	POSITIVE NAND GATE	.030	.060	14
3	7432	POSITIVE OR GATE	.345	.570	14
3	74S471	256×8 PROM	1.65	2.325	20
4	82S208	256×8 RAM (SIGNETICS)	2.700	3.700	22
8	74LS163A	SYNCRONOUS 4 bit COUNTERS	.760	1.280	16
2	74S138	DEMULTIPLEXER (3 TO 8)	.490	.740	16
6	82S191	2048×8 PROM (SIGNETICS) Tac=80 NS	3.900	5.250	24
2	93464	1024×8 ROM FAIRCHILD Tac=45 NS	1.100	1.500	24
1	DAC80	BURR BROWN 8 bit DAC	.900	.900	16
1	MDAC	COMPUTER LABS MULTIPLYING DAC 10 MHz	1.800	1.800	N/A
1	ADC-SH4B	ADC 4 bits WITH S/H	1.375	1.375	N/A
1	3554	OP AMP BURR BROWN	3.025	3.025	N/A
2	?	AMI 1024×4 VMOS RAM TAC=55 NS	1.200	1.800	18
4	74LS193	SYNCHRONOUS 4 bit COUNTER WITH CLEAR	.380	.680	16
4	74LS157	QUAD 2 TO 1 MULTIPLEXER	.194	.320	16
1	74L154	4 TO 16 LINE DECODER	.085	.140	24
TOTALS 151			39.355	56.307	

Figure 55. Part inventory and power consumption.

REFERENCES

1. *Intel Series 3000 Reference Manual,* Publication 98, 221A, 1975 Intel Corp.

2. *Series 3000 Cross Microprogramming System CROMIS Reference Specification.* Intel Corp., 1975 Revision 4.

3. Duda, R. O. and Hart, P.E., *Pattern Classification and Scene Analysis,* John Wiley and Sons, 1973, New York.

4. Sebestyen, G. S. *Decision-Making Processes in Pattern Recognition,* Macmillan Company, New York, 1962.

5. Sims, S.R.F., *Optical Contrast Variable Gate Centroid Tracker for the Intel 3000 Microprocessor,* TE-77-13, June 1977.

6. Lyman, J., *Growing Pin Count Is Forcing LSI Package Changes,* Electronics, March 19, 1977, McGraw Hill, pages 81-90.

APPENDIX A

Macrocode

(LISTING #1)

```
*
*
* SSW 5 ON FOR CENTROID TRACKING
*
* SSW 10 ON FOR SWITCH TO CORRELATOR WHILE CENTROID TRACKER
* IS ON
*
* SSW 10 OFF SWITCHES BACK TO CAGE MODE ON CENTROID TRACKER
* IF SSW 5 IS OFF
*
* IF SENSE SWITCH 7 IS ON .. YOU WILL GO TO THE AGC SCAN
* LOOP IN EITHER  CENTROID OR CORRELATION
*
*
*
*
   1: 00000 000002 100000                OCT    50
   2: 00001 000004 000000                OCT    100
   3: 00002 000001 000002      DELX      BSS    1
   4: 00003 000001 000003      DELY      BSS    1
   5: 00004 000001 000004      ASUM     BSS    1
   6: 00005 000001 000005      MEAN2    BSS    1
   7: 00006 000001 000006      XEEG     BSS    1
   8: 00007 000001 000007      XAEG     BSS    1
   9: 00010 000001 000010      XBLG2    BSS    1
  10: 00011 000001 000011      XALG     BSS    1
  11: 00012 000001 000012      YEEG     BSS    1
  12: 00013 000001 000013      YAEG     BSS    1
  13: 00014 000001 000014      YBLG2    BSS    1
  14: 00015 000001 000015      YALG     BSS    1
  15: 00016 000001 000016      XBAR     BSS    1
  16: 00017 000001 000017      YBAR     BSS    1
  17: 00020 000001 000020      BWF       BSS    1
                                        ABS    100
  19: 00100 125400 000356      CAGE      START  INIT
  20: 00101 000000 000367               LA     TEN
  21: 00102 176400 060003                DO     6
  22: 00103 124400 000000                WINDOW
  23: 00104 020000 000705                L      SAGC
  24: 00105 000000 000752                LA     FIFTY
  25: 00106 000400 000030                STA NX
  26: 00107 000400 000031                STA NY
  27: 00110 030000 000000                CLR
  28: 00111 000400 000032                STA    DELXX
  29: 00112 000400 000033                STA    DELYY
  30: 00113 000000 000401                LA     IPC
  31: 00114 000400 000020                STA    IP
  32: 00115 000000 000430                LA     C36
  33: 00116 176400 060004                DO     6
  34: 00117 020000 000771      AGCS      L      SCANN
```

```
00120 077400 000000                     LAIBUS
00121 005400 000766                     AND     SSW7
00122 006400 000000                     SZ
00123 010000 000117                     J       AGCS
00124 000000 000037                     LA      BIAS
00125 000400 000404                     STA     SBIAS
00126 000000 000036                     LA      GAIN
00127 000400 000405                     STA     SGAIN
00130 125400 000356                     START   INIT
00131 124400 000000         CAGE1       WINDOW
00132 077400 000000                     LAIBUS
00133 005400 000766                     AND     SSW7
00134 006400 000000                     SZ
00135 010000 000100                     J       CAGE
00136 020000 001003                     L       ZERR
00137 000000 000374                     LA      S62
00140 000400 000346                     STA     CX
00141 000400 000347                     STA     CY
00142 000000 000400                     LA      C36
00143 004400 000372                     OR      TARGET
00144 176400 060000                     DO      6
00145 125000 000000                     FRAMEW
00146 000000 000377                     LA      C33
00147 004400 000372                     OR      TARGET
00150 176400 060000                     DO      6
00151 120400 000350                     MEAN    ECOUNT
00152 174400 000000                    EQ
00153 000000 000350                     LA      ECOUNT
00154 032400 070000                     ALS     7
00155 000400 000350                     STA     ECOUNT
00156 030000 000000                     CLR
00157 174400 000000                    EQ
00160 003400 000350                     D       ECOUNT
00161 033000 140000                     LRS     14
00162 006000 000366                     C       SEVEN
00163 072000 000000                     SGE
00164 010000 000171                    J        BLACK
00165 030000 000000                     CLR
00166 000400 000372                     STA     TARGET
00167 000400 000020                     STA     BWF
00170 010000 000175                     J       WHITE
00171 000000 000371         BLACK      LA       OONE
00172 000400 000020                     STA     BWF
00173 000000 000370                     LA      TWENTY
00174 000400 000372                     STA     TARGET
00175 125400 000356         WHITE       START   INIT
*       STORE MODE LATCH FOR VIDEO
00176 000000 000364         START      LA       SIX
00177 004400 000372                     OR      TARGET
00200 176400 060000                    DO      6
```

```
 84: 00201 030000 000404            LA      SBIAS
 85: 00202 176400 040000            DO      4
 86: 00203 000000 000405            LA      SGAIN
 87: 00204 176400 050000            DO      5
*    REQUEST FRAME AND PROCESS CORRELATOR AGC GAIN AND BIAS
 88: 00205 125000 000000            FRAMEW
*         SET MODE LATCH FOR READ MODE
 89: 00206 000000 000362            LA      ONE
 90: 00207 004400 000372            OR      TARGET
 91: 00210 176400 060000            DO      6
 92: 00211 124400 000000            WINDOW
 93: 00212 077400 000000            LAIBUS
 94: 00213 005400 000376            AND     SSW1
 95: 00214 007000 000000            SNZ
 96: 00215 010000 000341            J       TT
 97: 00216 121000 000351            SAVEIJ  I
 98: 00217 120400 000350            MEAN    ECOUNT
 99: 00220 174400 000000            EQ
100: 00221 000000 000350            LA      ECOUNT
101: 00222 032400 070000            ALS     7
102: 00223 000400 000350            STA     ECOUNT
103: 00224 030000 000000            CLR
104: 00225 174400 000000            EQ
105: 00226 003400 000350            D       ECOUNT
106: 00227 033000 140000            LRS     14
107: 00230 000400 000005            STA     MEAN2
108: 00231 121400 000351            RESTIJ  I
109: 00232 122000 000002            EGATES  DELX
110: 00233 000000 000344            ADR     XG
111: 00234 000000 000346            ADR     CX
112: 00235 000000 000347            ADR     CY
113: 00236 121000 000351            SAVEIJ  I
114: 00237 123000 000007            CNTRD   XAEG
115: 00240 000000 000004            LA      ASUM
116: 00241 032400 070000            ALS     7
117: 00242 000400 000004            STA     ASUM
118: 00243 030000 000000            CLR
119: 00244 174400 000000            EQ
120: 00245 000000 000016            LA      XBAR
121: 00246 003400 000004            D       ASUM
122: 00247 000400 001012            STA     CXS
123: 00250 033000 140000            LRS     14
124: 00251 000400 000346            STA     CX
125: 00252 000000 000017            LA      YBAR
126: 00253 003400 000004            D       ASUM
127: 00254 000400 001011            STA     CYS
128: 00255 033000 140000            LRS     14
129: 00256 000400 000347            STA     CY
130: 00257 000000 001012            LA      CXS
131: 00260 033000 110000            LRS     11
```

```
00261 002400 001010                          S      0620
00262 020000 000406                          L      CONVET
00263 176400 070000                          DO     7
00264 000000 001011                          LA     CYS
00265 033000 110000                          LRS    11
00266 002400 001010                          S      0620
00267 020000 000406                          L      CONVET
00270 176400 100000                          DO     10
00271 000000 000003                          LA     DELY
00272 003000 000344                          M      XG
00273 174400 000000                          EQ
00274 000400 000353                          STA    AXGH
00275 000000 000002                          LA     DELX
00276 003000 000345                          M      YG
00277 174400 000000                          EQ
00300 000400 000354                          STA    AYGH
00301 121400 000351                          RESTIJ I
00302 122400 000353                          NGATEA AXGH
00303 030000 000002                          ADR    DELX
00304 174400 000000                          EQ
00305 000000 000353                          LA     AXGH
00306 032400 070000                          ALS    7
00307 000400 000353                          STA    AXGH
00310 030000 000000                          CLR
00311 174400 000000                          EQ
00312 003400 000353                          D      AXGH
00313 032000 140000                          ARS    14
00314 123000 000002                          NGATEB DELX
00315 000000 000346                          ADR    CX
00316 123400 000354                          NGATEC AYGH
00317 000000 000003                          ADR    DELY
00320 174400 000000                          EQ
00321 000000 000354                          LA     AYGH
00322 032400 070000                          ALS    7
00323 000400 000354                          STA    AYGH
00324 030000 000000                          CLR
00325 174400 000000                          EQ
00326 003400 000354                          D      AYGH
00327 032000 140000                          ARS    14
00330 124000 000003                          NGATEL DELY
00331 000000 000347                          ADR    CY
00332 077400 000000                          LAIBUS
00333 005400 000402                          AND    SSW10
00334 007000 000000                          SNZ
00335 010000 000176                          J      START
00336 000000 000401                          LA     IPC
00337 000400 000020                          STA    IP
00340 010000 000430                          J      START2
00341 000000 000372              TT          LA     TARGET
00342 004400 000367                          OR     TEN
```

```
182: 00343 010000 000131                      J        CAGE1
183: 00344 000001 000344        XG            BSS      1
184: 00345 000001 000345        YG            BSS      1
185: 00346 000001 000346        CX            BSS      1
186: 00347 000001 000347        CY            BSS      1
187: 00350 000001 000350        ECOUNT        BSS      1
188: 00351 000001 000351        I             BSS      1
189: 00352 000001 000352        J             BSS      1
190: 00353 000001 000353        AXGH          BSS      1
191: 00354 000001 000354        AYGH          BSS      1
192: 00355 000001 000355        AG2           BSS      1
193: 00356 000003 010000        INIT          OCT      61
194: 00357 000003 010000                      OCT      61
195: 00360 000003 030000                        OCT      63
196: 00361 000003 030000                        OCT      63
197: 00362 000000 010000        ONE           OCT      1
198: 00363 000000 020000        TWO           OCT      2
199: 00364 000000 060000        SIX             OCT      6
200: 00365 000000 030000        THREE         OCT      3
201: 00366 000000 070000        SEVEN           OCT      7
202: 00367 000000 100000        TEN             OCT      10
203: 00370 000001 000000        TWENTY        OCT      20
204: 00371 177777 170000        OONE            OCT      3777777
205: 00372 000001 000372        TARGET BSS    1
206: 00373 060000 000000        G               OCT      1400000
207: 00374 000003 020000        S62             OCT      62
208: 00375 077777 170000        CONN            OCT      1777777
209: 00376 000001 000000        SSW1            OCT      20
210: 00377 000000 130000        C33             OCT      13
211: 00400 000000 160000        C36             OCT      16
212: 00401 000055 160000        IPC             OCT      1336
213: 00402 000040 000000        SSW10           OCT      1000
214: 00403 000001 000403        SBWF            BSS      1
215: 00404 000001 000404        SBIAS           BSS      1
216: 00405 000001 000405        SGAIN           BSS      1
217: 00406 002000 000413        CONVET          A        BIAX
218: 00407 031400 000000                        TCA
219: 00410 002400 000362                        S        ONE
220: 00411 074400 000000                        RTN
221: 00412 000003 020000        O62             OCT      62
222: 00413 000040 000000        BIAX            OCT      1000
*
*
*         COMBINE BOTH TRACKERS.......
*
*
                                                ABS 20
224: 00020 000055 160000        IP              OCT      1336
225: 00021 000050 140000        RP              OCT      1214
226: 00022 000043 120000        CP              OCT      1072
```

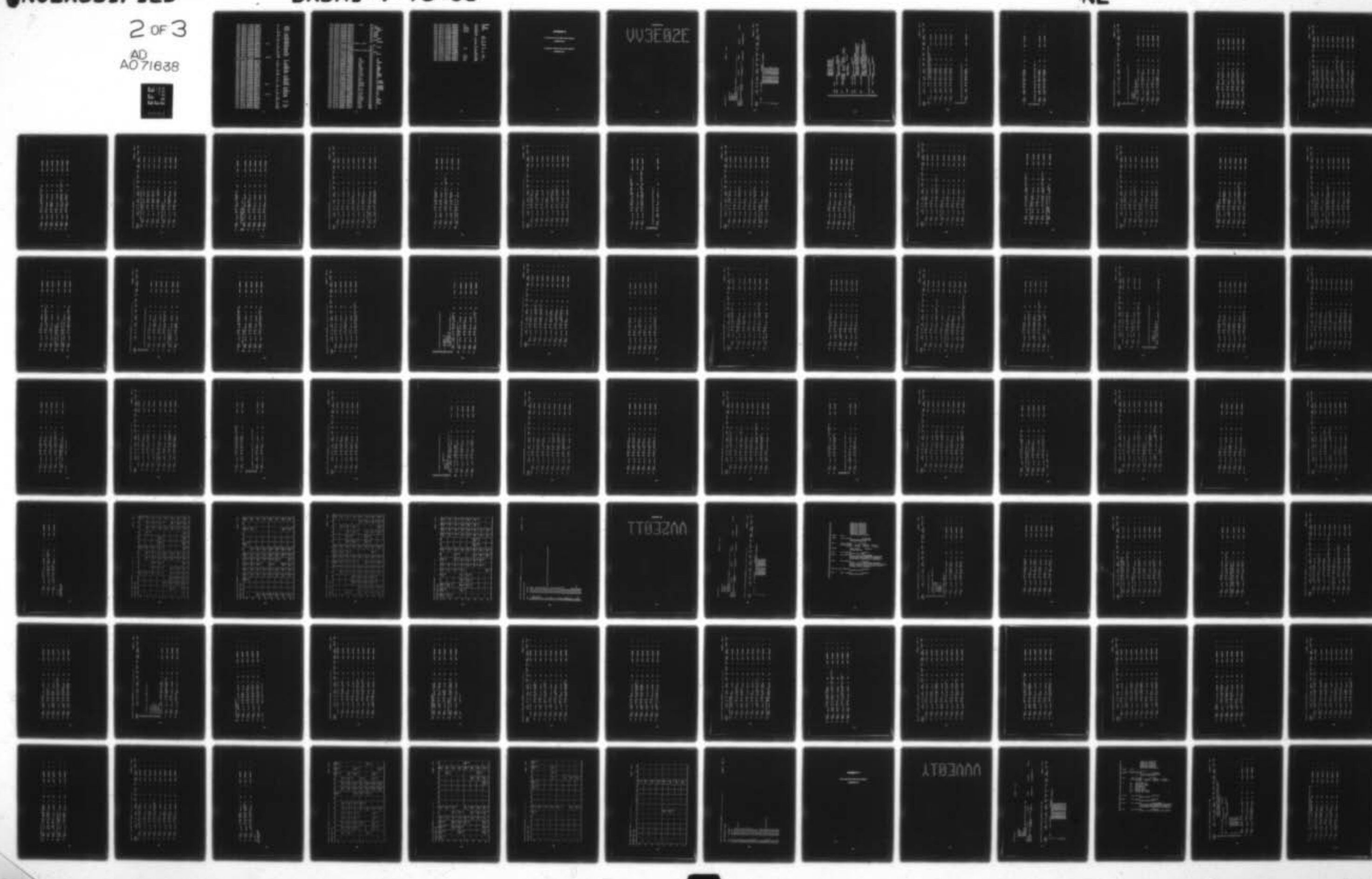
```
00023 000051 040000      IP2       OCT    1224
00024 000001 000024      MEAN      BSS    1
00025 000001 000025      STD       BSS    1
00026 000001 000026      MAXI      BSS    1
00027 000001 000027      MINI      BSS    1
00030 000001 000030      NX        BSS    1
00031 000001 000031      NY        BSS    1
00032 000001 000032      DELXX     BSS    1
00033 000001 000033      DELYY     BSS    1
00034 000001 000034      MODE      BSS    1
00035 000001 000035      BWC       BSS    1
00036 000001 000036      GAIN      BSS    1
00037 000001 000037      BIAS      BSS    1
00040 177766 070000      M153      OCT    3777547
00041 177766 170000      M145      OCT    3777557
                                   ABS    430
00430 000000 000346      START2    LA     CX
00431 000400 000031                STA    NY
00432 000000 000347                LA     CY
00433 000400 000030                STA    NX
00434 000000 000755                LA     ZERO
00435 000400 000032                STA    DELXX
00436 000400 000033                STA    DELYY
00437 000000 000362                LA     ONE
00440 000400 000035                STA    BWC
*      CROSS
00441 162000 000000                OCT    3440000
* BEGIN SCAN LOOP
* GO TO TRACK IF SSW7 OFF ELSE CONTINUE SCAN
00442 020000 000771      SCANNG    L      SCANN
00443 077400 000000                LAIBUS
00444 005400 000766                AND    SSW7
00445 006400 000000                SZ
00446 010000 000442                J      SCANNG
* GO TO TRACK MODE
* START TRACKING, STORE AGC PARAMETERS
00447 166412 160000                REF
00450 000000 000763                LA MEANO
00451 002000 000727                A      TP5
00452 000400 000760                STA    MNPK
00453 002400 000730                S      T1P5
00454 000400 000756                STA    MNSBPK
00455 000000 000764                LA     STDO
00456 002000 000734                A      TP5D
00457 000400 000761                STA    SDPK
00460 006000 000741                C      T13P5
00461 072400 000000                SLE
00462 000000 000731                LA     T14P0
00463 002400 000730                S      T1P5
00464 000400 000757                STA    SDSBPK
```

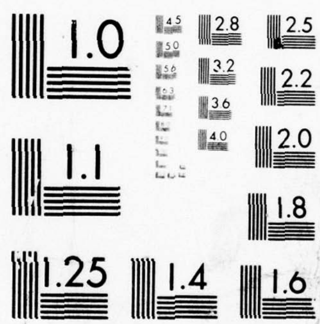
```
272: 00465 010000 000473                       J      OVER
273: 00466 164400 000000       LOOP            WAT
274: 00467 077400 000000                       LAIBUS
275: 00470 005400 000402                       AND    SSW10
276: 00471 007000 000000                       SNZ
277: 00472 010000 000100                       J      CAGE
278: 00473 165372 120000       OVER            IMG
279: 00474 000000 000037                       LA     BIAS
280: 00475 176400 040000                       DO     4
281: 00476 000000 000036                       LA     GAIN
282: 00477 176400 050000                       DO     5
283: 00500 164000 000000                       FRM
284: 00501 167000 000000                       COR
285: 00502 163024 160000                       MIN
* MOVE WINDOW TO NEW TRACKING POSITION
286: 00503 000000 000032                       LA     DELXX
287: 00504 002000 000030                       A      NX
288: 00505 000400 000030                       STA    NX
289: 00506 000000 000033                       LA     DELYY
290: 00507 002000 000031                       A      NY
291: 00510 000400 000031                       STA    NY
* BOUND ON NX AND NY
292: 00511 000000 000030                       LA     NX
293: 00512 006000 000732                       C      N10
294: 00513 072400 000000                       SLE
295: 00514 010000 000520                       J      NCK
296: 00515 000000 000732                       LA     N10
297: 00516 000400 000030                       STA    NX
298: 00517 010000 000525                       J      OUTH
299: 00520 006000 000733       NCK             C      N90
300: 00521 072000 000000                       SGE
301: 00522 010000 000525                       J      OUTH
302: 00523 000000 000733                       LA     N90
303: 00524 000400 000030                       STA    NX
304: 00525 000000 000031       OUTH            LA     NY
305: 00526 006000 000732                       C      N10
306: 00527 072400 000000                       SLE
307: 00530 010000 000534                       J      NCK2
308: 00531 000000 000732                       LA     N10
309: 00532 000400 000031                       STA    NY
310: 00533 010000 000541                       J      OUTH2
311: 00534 006000 000733       NCK2            C      N90
312: 00535 072000 000000                       SGE
313: 00536 010000 000541                       J      OUTH2
314: 00537 000000 000733                       LA     N90
315: 00540 000400 000031                       STA    NY
316: 00541 163400 000000       OUTH2           WIN
317: 00542 000000 000030                       LA     NX
318: 00543 002400 000412                       S      062
319: 00544 032400 030000                       ALS    3
```

```
00545 020000 000406               L      CONVET
00546 176400 100000               DO     10
00547 000000 000031               LA     NY
00550 002400 000412               S      062
00551 032400 030000               ALS    3
00552 020000 000406               L      CONVET
00553 176400 070000               DO     7
00554 020000 000564               L      AGC
00555 077400 000000               LAIBUS
00556 005400 000766               AND    SSW7
00557 007000 000000               SNZ
00560 010000 000563               J      OOOV
00561 164400 000000               WAT
00562 010000 000430               J      START2
00563 010000 000466       OOOV    J      LOOP
* SUBROUTINE AGC FOLLOWS
*          MNV
00564 166000 000000       AGC     OCT    3540000
00565 000000 000024               LA     MEAN
00566 003400 000744               D      T81
00567 032000 050000               ARS    5
00570 000400 000024               STA    MEAN
00571 161400 000000               OCT    3430000
00572 000000 000025               LA     STD
00573 003400 000745               D      T81PRM
00574 032000 020000               ARS    2
00575 000400 000025               STA    STD
* FILTER STANDARD DEVIATION
00576 000000 000764               LA     STDO
00577 003000 000750               M      C2
00600 000400 000762               STA    TEMP
00601 000000 000025               LA     STD
00602 003000 000747               M      ONEMC2
00603 002000 000762               A      TEMP
00604 000400 000764               STA    STDO
00605 006000 000757               C      SDSBPK
00606 072000 000000               SGE
00607 010000 000622               J      INCRE
00610 006000 000761               C      SDPK
00611 072000 000000               SGE
00612 010000 000647               J      NOCHG
00613 000000 000036               LA     GAIN
00614 003000 000751               M      GP933
00615 000400 000036               STA    GAIN
00616 000000 000037               LA     BIAS
00617 003000 000751               M      GP933
00620 000400 000037               STA    BIAS
00621 010000 000647               J      NOCHG
00622 000000 000036       INCRE   LA     GAIN
00623 006000 000753               C      FULLG
```

AD-A071 638 ARMY MISSILE RESEARCH AND DEVELOPMENT COMMAND REDSTO--ETC F/G 9/2
APPLICATION OF A MICROPROGRAMMED, BIT SLICE MICROPROCESSOR TO T--ETC(U)
SEP 78 L G MINOR
UNCLASSIFIED DRDMI-T-78-53 NL

2 OF 3
AD
A071638





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

```
00624 071000 000000                  SL
00625 010000 000647                  J      NOCHG
00626 003400 000751                  D      GP933
00627 006000 000753                  C      FULLG
00630 072400 000000                  SLE
00631 010000 000637                  J      FULL
00632 000400 000036                  STA    GAIN
00633 000000 000037                  LA     BIAS
00634 003400 000751                  D      GP933
00635 000400 000037                  STA    BIAS
00636 010000 000647                  J      NOCHG
00637 000000 000036      FULL        LA     GAIN
00640 003400 000753                  D      FULLG
00641 000400 000762                  STA    TEMP
00642 000000 000037                  LA     BIAS
00643 003400 000762                  D      TEMP
00644 000400 000037                  STA    BIAS
00645 000000 000753                  LA     FULLG
00646 000400 000036                  STA    GAIN
00647 037000 000000      NOCHG       NOP
* FILTER MEAN VALUE
00650 000000 000763                  LA     MEANO
00651 003000 000765                  M      C1
00652 000400 000762                  STA    TEMP
00653 000000 000024                  LA     MEAN
00654 003000 000767                  M      ONEMC1
00655 002000 000762                  A      TEMP
00656 000400 000763                  STA    MEANO
00657 006000 000760                  C      MNPK
00660 072000 000000                  SGE
00661 010000 000666                  J      NEXT
00662 000000 000037                  LA     BIAS
00663 002000 000746                  A      ONEONE
00664 000400 000037                  STA    BIAS
00665 010000 000674                  J      NEXT2
00666 006000 000756      NEXT        C      MNSBPK
00667 072400 000000                  SLE
00670 010000 000674                  J      NEXT2
00671 000000 000037                  LA     BIAS
00672 002400 000746                  S      ONEONE
00673 000400 000037                  STA    BIAS
00674 000000 000037      NEXT2       LA     BIAS
00675 006000 000755                  C      ZERO
00676 072000 000000                  SGE
00677 030000 000000                  CLR
00700 006000 000754                  C      FULLB
00701 072400 000000                  SLE
00702 000000 000754                  LA     FULLB
00703 000400 000037                  STA    BIAS
00704 074400 000000                  RTN
```

```
416: 00705 037000 000000      SAGC        NOP
417: 00706 000000 000736                  LA T7P0
418: 00707 000400 000756                  STA MNSBPK
419: 00710 000400 000763                  STA    MEANO
420: 00711 000000 000740                  LA T14P5
421: 00712 000400 000761                  STA SDPK
422: 00713 000400 000764                  STA STDO
423: 00714 000000 000737                  LA T8P0
424: 00715 000400 000760                  STA    MNPK
425: 00716 000000 000741                  LA T13P5
426: 00717 000400 000757                  STA    SDSBPK
* INITALIZE GAIN TO APPROXIMATELY UNITY
427: 00720 000000 000742                  LA T31
428: 00721 000400 000036                  STA    GAIN
429: 00722 176400 050000                  DO 5
* INITALIZE BIAS TO APPROXIMATELY BLACK LEVEL
430: 00723 000000 000743                  LA DCBLK
431: 00724 000400 000037                  STA    BIAS
432: 00725 176400 040000                  DO 4
433: 00726 074400 000000                  RTN
434: 00727 000001 000000      TP5         OCT    20
435: 00730 000003 000000      T1P5        OCT    60
436: 00731 001600 000000      T14P0       OCT    34000
437: 00732 000000 120000      N10         OCT    12
438: 00733 000005 120000      N90         OCT    132
439: 00734 000040 000000      TP5D        OCT    1000
440: 00735 000140 000000      T1P5D       OCT    3000
441: 00736 000016 000000      T7P0        OCT    340
442: 00737 000020 000000      T8P0        OCT    400
443: 00740 001640 000000      T14P5       OCT    35000
444: 00741 001540 000000      T13P5       OCT    33000
445: 00742 000001 110000      T31         OCT    31
446: 00743 000000 060000      DCBLK       OCT    6
447: 00744 005040 000000      T81         OCT    121000
448: 00745 050400 000000      T81PRM      OCT    1210000
449: 00746 000000 010000      ONEONE      OCT    1
450: 00747 040000 000000      ONEMC2      OCT    1000000
451: 00750 040000 000000      C2          OCT    1000000
452: 00751 075660 000000      GP933       OCT    1735400
453: 00752 000003 020000      FIFTY       OCT    62
454: 00753 000017 170000      FULLG       OCT    377
455: 00754 000017 170000      FULLB       OCT    377
456: 00755 000000 000000      ZERO        OCT    0
457: 00756 000016 000000      MNSBPK      OCT    340
458: 00757 001540 000000      SDSBPK      OCT    33000
459: 00760 000020 000000      MNPK        OCT    400
460: 00761 001640 000000      SDPK        OCT    35000
461: 00762 000001 000762      TEMP        BSS    1
462: 00763 000001 000763      MEANO       BSS    1
463: 00764 000001 000764      STDO        BSS    1
```

```
00765 060000 000000     C1        OCT    1400000
00766 000004 000000     SSW7      OCT    100
00767 020000 000000     ONEMC1    OCT    400000
00770 000001 160000     FRMRG     OCT    36
00771 164000 000000     SCANN     FRM
00772 164400 000000               WAT
00773 165006 170000               IMG
00774 020000 000564               L      AGC
00775 000000 000037               LA     BIAS
00776 176400 040000               DO     4
00777 000000 000036               LA     GAIN
01000 176400 050000               DO     5
01001 020000 001003               L      ZERR
01002 074400 000000               RTN
01003 000000 001007     ZERR      LA     C777
01004 176400 070000               DO     7
01005 176400 100000               DO     10
01006 074400 000000               RTN
01007 000037 170000     C777      OCT    777
01010 000031 000000     0620      OCT    620
01011 000001 001011     CYS       BSS    1
01012 000001 001012     CXS       BSS    1
```

APPENDIX B

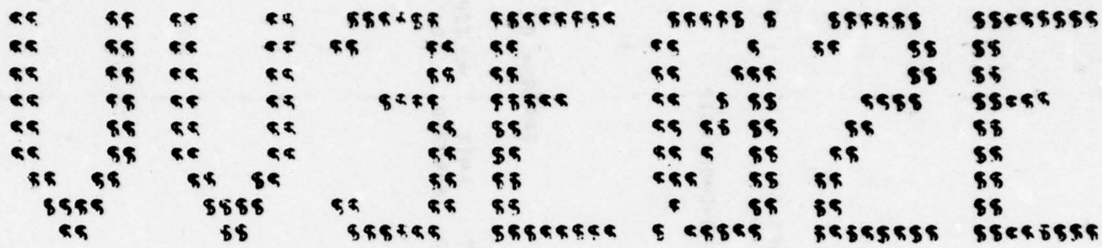
Correlation Tracker Microcode (Page 2)

(LISTING #2)

Correlation Tracker Microcode (Page 3)

(LISTING #3)

(LISTING #2)



```
XMAS VERS 2.0                                                            ERRORS= 0    PAGE 1

        $LISTFILE=2
        $BITS
        $CROSSREF
        $IMAGE
        $WIDTH=132
        $TITLE='CORRELATION MICROCODE ..............................'
        $DISPLAY(ALL)
BITS=1           LEFT=1             FORMS=0            IMAGE=1            LINES=60           PRINT=1
RIGHT=72
TITLE='CORRELATION MICROCODE ..............................'
WIDTH=132        CROSSREF=1         LISTFILE=2         SOURCEFILE=2       MICROMEMORY=512



XMAS VERS 2.0 CORRELATION MICROCODE ..............................                 ERRORS= 0    PAGE 2

RECORD          CPE     FI  FO    JUMP     KBUSS    PAGEF  CBUS  PAUSEF  STRB  LPCNT    EMIT    MULTIPL  SPRF
NUMBER       6543210    10  10  6543210  76543210    10    210     0      0    43210  76543210     0      0

    1
    2  KBUSS        FIELD  LENGTH=8            DEFAULT=0
    3                      MICROPS(K00000=00H K00001=01H K00002=02H
    4                      K00003=03H K00004=04H
    5                      K00005=05H K00006=06H
    6                      K00007=07H K00008=08H
    7                      K00009=09H K0000A=0AH
    8                      K0000B=0BH K0000C=0CH
    9                      K0000D=0DH  K0000E=0EH
   10                      K0000F=0FH K00010=10H
   11                      K00011=11H K00012=12H
   12                      K00013=13H K00014=14H
   13                      K00015=15H K00016=16H
   14                      K00017=17H K00018=18H
   15                      K00019=19H K0001A=1AH
```

```
                             K0001B=1BH K0001C=1CH
                             K0001D=1DH K0001E=1EH
                             K0001F=1FH K01FFF=3FH
                             K80000=80H K7FFFF=7FH
                             KFFFFF=377Q KFFFFA=0FAH);
KBUSS      KBUS;

PAGEF      FIELD LENGTH=2                DEFAULT=1B
                             MICROPS(PAGE1=00B PAGE2=01B
                             PAGE3=10B PAGE4=11B);

CBUS       FIELD  LENGTH=3          DEFAULT=0
         MICROPS(NBO=000B   INH=001B   RMW=010B   CNB=011B
                 RTN=100B   ROT=101B   RRM=110B   RWM=111B);

PAUSEF                       FIELD LENGTH=1        DEFAULT=1
                             MICROPS(PAUSE=0);


STRB       FIELD LENGTH=1                DEFAULT=0
                             MICROPS(STROBE=1B);
LPCNT      FIELD LENGTH=5                DEFAULT=00010B
                             MICROPS(C0=00010B LCNT1=00100B LCNT2=01000B
                             LCNT3=01100B LCNT4=10000B ITCNT1=00111B
                             ITCNT2=01011B ITCNT3=01111B ITCNT4=10011B);

EMIT       FIELD LENGTH=8                DEFAULT=0
                             MICROPS ( M1=0FFH M2=0FEH M3=0FDH M4=0FCH
                             M5=0FBH M6=0FAH M7=0F9H M8=0F8H M9=0F7H M10=0F6H
                             M11=0F5H M12=0F4H M13=0F3H M14=0F2H M15=0F1H
                             M16=0F0H M17=0EFH M18=0EEH);

MULTIPLY FIELD LENGTH=1                  DEFAULT=0
                             MICROPS(MULT=1B);

SPRF       FIELD  LENGTH=1               DEFAULT=0
                             MICROPS(SPARE=01H);
```

```
XMAS VERS 2.0 CORRELATION MICROCODE ............................                ERRORS= 0   PAGE 3

RECORD               CPF     FI  FO    JUMP      KRUSS     PAGEF  CBUS  PAUSFF  STRB  LPCNT    EMIT     MULTIPL  SPRF
NUMBER             6543210   10  10  6543210   76543210    10     210     0      0    43210  76543210     0       0

  55
  56   05EH!                 NOP(AC) JPR(NOT1,NOT2,NOT3,STD,CROSS,NOT6,MIN,WIN,FRM,
  57                                       WAT,IMG,NOT12,MNV,REF,CORR,NOT16);
       (005EH)     1101101   11  00  1100111   00000000    01     000     1      0    00010  00000000     0       0

  58   070H!        NOT1!    NOP(AC) PAGE1 JZR(FETCH);
       (0070H)     1101101   11  00  0101111   00000000    00     000     1      0    00010  00000000     0       0

  59   071H!        NOT2!    NOP(AC) PAGE1 JZR(FETCH);
       (0071H)     1101101   11  00  0101111   00000000    00     000     1      0    00010  00000000     0       0

  60   072H!        NOT3!    NOP(AC) PAGE1 JZR(FETCH);
       (0072H)     1101101   11  00  0101111   00000000    00     000     1      0    00010  00000000     0       0

  61   075H!        NOT6!    NOP(AC) PAGE1 JZR(FETCH);
       (0075H)     1101101   11  00  0101111   00000000    00     000     1      0    00010  00000000     0       0

  62
  63
  64
  65
  66
  67   /* ROUTINE MIN MIN MIN MIN MIN MIN MIN MIN MIN */
  68
  69
  70   076H!        MIN!     NOP(AC) PAGE3 JCR(MIN);
       (0076H)     1101101   11  00  0110110   00000000    10     000     1      0    00010  00000000     0       0
```

```
07BH!       NOTI2! NOP(AC) PAGE1 JZR(FETCH);
(007BH)  1101101  11  00  0101111  00000000   00     000     1     0    00010  00000000    0    0


/* ROUTINE CORR CORR CORR CORR CORR CORR CORR CORR  */


07EH!       CORR!  NOP(AC) PAGE3 JCR(CORR);
(007EH)  1101101  11  00  0111110  00000000   10     000     1     0    00010  00000000    0    0

07FH!       NOTI6! NOP(AC) PAGE1 JZR(FETCH);
(007FH)  1101101  11  00  0101111  00000000   00     000     1     0    00010  00000000    0    0
```

```
XMAS VERS 2.0 CORRELATION MICROCODE ..............................                     ERRORS= 0   PAGE 4

RECORD              CPE     FT  FO   JUMP     KBUSS    PAGEF  CBUS  PAUSEF  STRB  LPCNT   EMIT     MULTIPL  SPRF
NUMBER            6543210   10  10  6543210  76543210   10    210     0      0    43210  76543210    0       0

  82
  83
  84
  85
  86
  87  /* IMG IMG IMO IMG IMG IMG IMO IMG IMG IMG IMG  */
  88
  89
  90     /* R7=93*/
  91     /*R8=SA         */
  92     /*R9=STA */
  93     /*  FORM STARTING ADDRESS */
  94  07AH!      IMG!   CLR(AC) ;
      (007AH)  1001101  11  00  0001000  00000000   01    000     1      0    00010  00000000    0       0

  95  08AH!             LMI(AC) K00019 RRM ;
      (008AH)  0011101  11  00  0111111  00011001   01    110     1      0    00010  00000000    0       0

  96  08FH!  ACM(T) ;  /*T=NY */
      (008FH)  0001010  11  00  0001001  00000000   01    000     1      0    00010  00000000    0       0

  97  09FH!             CLR(AC);
      (009FH)  1001101  11  00  0001010  00000000   01    000     1      0    00010  00000000    0       0

  98  0AFH!             LMI(AC) K00009 ;
      (00AFH)  0011101  11  00  0001011  00001001   01    000     1      0    00010  00000000    0       0
```

```
99  0BFH!                    CMR(AC) ;
    (00BFH)  1111101  11  00  0001100  00000000   01   000   1   0   00010  00000000   0   0

100 0CFH!                    ALR(T) FF1 ;  /*NY-9 TO T, AC  */
    (00CFH)  0001100  11  11  0001101  11111111   01   000   1   0   00010  00000000   0   0

101 0DFH!                    CLR(T) ;
    (00DFH)  1001100  11  00  0001110  00000000   01   000   1   0   00010  00000000   0   0

102 0EFH!                    ADR(AC) ;  /*2(NY-9) TO AC */
    (00EFH)  0111101  11  00  0001111  11111111   01   000   1   0   00010  00000000   0   0

103 0FFH!                    ADR(AC) ;  /* 4(NY-9) TO AC */
    (00FFH)  0111101  11  00  0010000  11111111   01   000   1   0   00010  00000000   0   0

104 10FH!                    ADR(T) ;  /*AC TO T  */
    (010FH)  0111100  11  00  0010001  11111111   01   000   1   0   00010  00000000   0   0
```

```
XMAS VERS 2.0 CORRELATION MICROCODE ...............................                ERRORS= 0   PAGE 5

RECORD              CPE     FI FO   JUMP     KBUSS    PAGEF CBUS PAUSEF STRB LPCNT    EMIT    MULTIPL SPRF
NUMBER            6543210   10 10  6543210  76543210   10   210    0     0   43210  76543210     0      0

 105  11FH!                 ADR(AC) ;  /*8(NY-9) TO AC */
      (011FH)     0111101   11 00  0010010  11111111   01   000    1     0   00010  00000000     0      0

 106  12FH!                 ADR(AC) ;  /*16(NY-9) TO AC  */
      (012FH)     0111101   11 00  0010011  11111111   01   000    1     0   00010  00000000     0      0

 107  13FH!                 ADR(AC) ;  /*32(NY-9) TO AC */
      (013FH)     0111101   11 00  0010100  11111111   01   000    1     0   00010  00000000     0      0

 108  14FH!                 ADR(T) ; /*AC+T TO T */
      (014FH)     0111100   11 00  0010101  11111111   01   000    1     0   00010  00000000     0      0

 109  15FH!                 ADR(AC)              ;  /*64(NY-9) TO AC */
      (015FH)     0111101   11 00  0010110  11111111   01   000    1     0   00010  00000000     0      0

 110  16FH!                 ALR(T) ;  /*AC+T TO T+AC=100(NY-9) */
      (016FH)     0001100   11 00  0010111  11111111   01   000    1     0   00010  00000000     0      0

 111  17FH!                 CLR(T) ;
      (017FH)     1001100   11 00  0011000  00000000   01   000    1     0   00010  00000000     0      0

 112  18FH!                 LMI(T) K00018 RRM ;
      (018FH)     0011100   11 00  0011001  00011000   01   110    1     0   00010  00000000     0      0
```

```
113 19FH!                   AMA(AC) ;  /*100(NY-9)+NX TO AC */
    (019FH)  0001011  11  00  0011010  11111111  01   000   1   0   00010  00000000   0   0

114 1AFH!                   CLR(T) ;
    (01AFH)  1001100  11  00  0000001  00000000  01   000   1   0   00010  00000000   0   0

115 01FH!                   LMI(T) K00009 ;
    (001FH)  0011100  11  00  0000010  00001001  01   000   1   0   00010  00000000   0   0

116 02FH!                   CMR(T) ;
    (002FH)  1111100  11  00  0000011  00000000  01   000   1   0   00010  00000000   0   0

117 03FH!                   ALR(T) FF1 ;  /*100(NY-9) + NX-9 TO AC.T */
    (003FH)  0001100  11  11  0000100  11111111  01   000   1   0   00010  00000000   0   0

118 04FH!                   SDR(R8) FF1 ;  /*SA=AC */
    (004FH)  0101000  11  11  0000101  11111111  01   000   1   0   00010  00000000   0   0
```

```
XMAS VERS 2.0 CORRELATION MICROCODE ..............................                         ERRORS= 0   PAGE 6

RECORD             CPE     FI FO   JUMP     KBUSS    PAGEF CBUS PAUSFF STRB LPCNT   EMIT     MULTIPL SPRF
NUMBER           6543210   10 10 6543210  76543210    10   210    0     0   43210 76543210      0      0

 119      /*FORM CONSTANT R7=83 AND LOAD COUNTERS  */
 120  05FH!                CLR(R7) ;
      (005FH)  1000111   11 00 0000110  00000000    01   000    1     0   00010 00000000      0      0

 121  06FH!                LMI(R7) K0001F LCNT1 M1B ;
      (006FH)  0010111   11 00 0111110  00011111    01   000    1     0   00100 11101110      0      0

 122  06EH!                LMI(R7) K0001F LCNT2 M1B ;
      (006EH)  0010111   11 00 0000001  00011111    01   000    1     0   01000 11101110      0      0

 123  01EH!                LMI(R7) K00015 ;   /*R7=83 */
      (001EH)  0010111   11 00 0000010  00010101    01   000    1     0   00010 00000000      0      0

 124      /* FETCH IMAGE POINTER */
 125  02EH!                CLR(T) ;
      (002EH)  1001100   11 00 0000011  00000000    01   000    1     0   00010 00000000      0      0

 126  03EH!                LMI(T) K00010 RRM ;
      (003EH)  0011100   11 00 0111101  00010000    01   110    1     0   00010 00000000      0      0

 127  03DH!                ACM(AC) ;
      (003DH)  0001011   11 00 0000001  00000000    01   000    1     0   00010 00000000      0      0
```

```
128  01DH!               SDR(R9) FF1 ;  /*R9=IP=STA */
     (001DH)  0101001  11  11  0000010  11111111  01   000   1   0   00010  00000000   0   0

129      /* GO TO READ MODE */
130       /* UPDATE  MODE  STATUS   WORD  */
131  02DH!               CLR(T) ;
     (002DH)  1001100  11  00  0000100  00000000  01   000   1   0   00010  00000000   0   0

132  04DH!               LMI(T) K0001C RRM ;
     (004DH)  0011100  11  00  0000101  00011100  01   110   1   0   00010  00000000   0   0

133  05DH!               ACM(AC) ;
     (005DH)  0001011  11  00  0000110  00000000  01   000   1   0   00010  00000000   0   0

134  06DH!               CLR(T) ;
     (006DH)  1001100  11  00  0011111  00000000  01   000   1   0   00010  00000000   0   0

135  1FDH!               INR(T) FF1 ;
     (01FDH)  0111100  11  11  0001000  00000000  01   000   1   0   00010  00000000   0   0
```

```
XMAS VERS 2.0 CORRELATION MICROCODE ............................          ERRORS= 0   PAGE 7

RECORD                CPE     FI  FO   JUMP     KBUSS    PAGEF  CBUS  PAUSEF  STRB  LPCNT    EMIT    MULTIPL  SPRF
NUMBER              6543210   10  10  6543210  76543210   10    210     0      0    43210  76543210    0       0

 136  08DH!                   ORR(T) :
       (008DH)      1101100   11  00  0001001  11111111   01    000     1      0    00010  00000000    0       0

 137  09DH!                   ILR(T) :
       (009DH)      0001100   11  00  0001110  00000000   01    000     1      0    00010  00000000    0       0

 138  0EDH!                   CLR(T) :
       (00EDH)      1001100   11  00  0001111  00000000   01    000     1      0    00010  00000000    0       0

 139  0FDH!                   LMI(T) K0001C RWM :
       (00FDH)      0011100   11  00  0010001  00011100   01    111     1      0    00010  00000000    0       0

 140  11DH!                   CLR(T) :
       (011DH)      1001100   11  00  0010010  00000000   01    000     1      0    00010  00000000    0       0

 141  12DH!                   LMI(T) K00006 ROT :
       (012DH)      0011100   11  00  0111010  00000110   01    101     1      0    00010  00000000    0       0

 142        /* BEGIN LOOPS    */
 143  12AH!          LP1!     LMI(RB) FF1 RIN :
       (012AH)      0011000   11  11  0010011  00000000   01    100     1      0    00010  00000000    0       0

 144  13AH!          LP1X!    LDI(AC) K0000F ITCNT1 FF1:
       (013AH)      0101111   11  11  0111100  00001111   01    000     1      0    00111  00000000    0       0
```

```
145  13CH!              LMI(R9) FF1 RWM JFL(LP1,CT1) ;
     (013CH)  0011001   11  11  1000010  00000000  01   111   1   0   00010  00000000   0   0

146  12BH!      CT1!    ILR(R7) ITCNT2 ;
     (012BH)  0000111   11  00  0111100  00000000  01   000   1   0   01011  00000000   0   0

147  12CH!              ADR(R8) LCNT1 M18 JFL(LP2,CT2) ;  /*SA=SA+R3  */
     (012CH)  0111000   11  00  1000101  11111111  01   000   1   0   00100  11101110   0   0

148  15AH!      LP2!    LMI(R8) FF1 RIN JMP(LP1X) ;
     (015AH)  0011000   11  11  0010011  00000000  01   100   1   0   00010  00000000   0   0

149     /* GO TO STORE MODE WITH NO REQ    */
150  15BH!      CT2!    CLR(AC) ;
     (015BH)  1001101   11  00  0111100  00000000  01   000   1   0   00010  00000000   0   0
```

```
RECORD              CPF     FI FO   JUMP     KBUSS    PAGEF  CBUS  PAUSFF  STRB  LPCNT    EMIT     MULTIPL  SPRF
NUMBER            6543210  10 10  6543210  76543210    10    210     0      0    43210  76543210     0       0

 151   15CH!                LMI(AC) K0001C RRM ;
       (015CH)    0011101  11 00  0010110  00011100    01    110     1      0    00010  00000000     0       0

 152   16CH!                ACM(AC) ;  /*AC=MODE  */
       (016CH)    0001011  11 00  0010111  00000000    01    000     1      0    00010  00000000     0       0

 153   17CH!                CLR(T) ;
       (017CH)    1001100  11 00  0011000  00000000    01    000     1      0    00010  00000000     0       0

 154   18CH!                LMI(T) K00003 ;
       (018CH)    0011100  11 00  0011001  00000011    01    000     1      0    00010  00000000     0       0

 155   19CH!                CMR(T) ;  /*T=KFFFFC  */
       (019CH)    1111100  11 00  0011010  00000000    01    000     1      0    00010  00000000     0       0

 156   1ACH!                ANR(T) ;  /*T=T.AND.KFFFFC  */
       (01ACH)    1001100  11 00  0011011  11111111    01    000     1      0    00010  00000000     0       0

 157   1BCH!                ILR(T) ;
       (01BCH)    0001100  11 00  0011100  00000000    01    000     1      0    00010  00000000     0       0

 158   1CCH!                CLR(T) ;
       (01CCH)    1001100  11 00  0011101  00000000    01    000     1      0    00010  00000000     0       0
```

```
1DCH!                LMI(T) K0001C RWM ;  /*MODE=MODE.AND.KFFFFC  */
(01DCH)  0011100  11  00  0111000  00011100   01    111    1     0    00010  00000000    0     0

1D8H!                CLR(T) ;
(01D8H)  1001100  11  00  0011110  00000000   01    000    1     0    00010  00000000    0     0

1F8H!                LMI(T) K00006 ROT PAGE1 JZR(FETCH) ;  /*MODFLAT=MODE *
(01F8H)  0011100  11  00  0101111  00000110   00    111    1     0    00010  00000000    0     0





/* ROUTINE WIN WIN WIN WIN WIN WIN WIN WIN WIN  */


077H!       WIN!     CLR(T ) ;
(0077H)  1001100  11  00  0001001  00000000   01    000    1     0    00010  00000000    0     0
```

```
XMAS VERS 2.0 CORRELATION MICROCODE ..............................         ERRORS= 0    PAGE 9

RECORD              CPF     FI FO   JUMP     KRUSS    PAGEF  CBUS  PAUSEF  STRB  LPCNT   EMIT     MULTIPL  SPRF
NUMBER            6543210   10 10  6543210  76543210   10    210     0      0    43210  76543210    0       0

 171  097H!                 LMI(T ) K00018 FF1 RRM ;
         (0097H)  0011100   11 11  0111001  00011000   01    110     1      0    00010  00000000    0       0

 172  099H!                 ACM(AC) ;
         (0099H)  0001011   11 00  0001010  00000000   01    000     1      0    00010  00000000    0       0

 173  0A9H!                 SDR(R5) FF1 ;  /*R5=C(18)=NX*/
         (00A9H)  0100101   11 11  0001011  11111111   01    000     1      0    00010  00000000    0       0

 174  0B9H!                 LMI(T ) RRM ;
         (00B9H)  0011100   11 00  0001100  00000000   01    110     1      0    00010  00000000    0       0

 175  0C9H!                 ACM(AC) ;
         (00C9H)  0001011   11 00  0001101  00000000   01    000     1      0    00010  00000000    0       0

 176  0D9H!                 SDR(R6) FF1 ;  /*R6=C(19)=NY*/
         (00D9H)  0100110   11 11  0001110  11111111   01    000     1      0    00010  00000000    0       0

 177  /*LOAD COLUMN LEFT LATCH  */
 178  0E9H!                 CLR(R7) ;
         (00E9H)  1000111   11 00  0001111  00000000   01    000     1      0    00010  00000000    0       0

 179  0F9H!                 LMI(R7) K00005 ;
         (00F9H)  0010111   11 00  0010000  00000101   01    000     1      0    00010  00000000    0       0
```

```
180  109H!            CMR(R7) ;
       (0109H)  1110111  11  00  0010001  00000000  01   000   1   0   00010  00000000   0   0

181  119H!            ILR(R5) ;
       (0119H)  0000101  11  00  0010010  00000000  01   000   1   0   00010  00000000   0   0

182  129H!            ALR(R7) FF1 ;
       (0129H)  0000111  11  11  0010011  11111111  01   000   1   0   00010  00000000   0   0

183  139H!            CLR(T) ;
       (0139H)  1001100  11  00  0010100  00000000  01   000   1   0   00010  00000000   0   0

184  149H!            LMI(T) FF1 ROT ; /*NX-5 TO COL L LATCH */
       (0149H)  0011100  11  11  0010101  00000000  01   101   1   0   00010  00000000   0   0

185  /* LOAD COLUMN RIGHT LATCH */
```

```
XMAS VERS 2.0 CORRELATION MICROCODE ..............................                  ERRORS= 0   PAGE 10

RECORD                CPE    FI  FO    JUMP      KBUSS    PAGEF  CBUS  PAUSEF  STRB  LPCNT    EMIT     MULTIPL  SPRF
NUMBER            6543210  10  10  6543210  76543210    10    210     0       0    43210  76543210     0       0

 186  159H!                LMI(R5) K00005 ;
      (0159H)     0010101  11  00  0010110  00000101    01    000     1       0    00010  00000000     0       0

 187  169H!                ILR(R5) ;
      (0169H)     0000101  11  00  0010111  00000000    01    000     1       0    00010  00000000     0       0

 188  179H!                LMI(T) FF1 ROT ;  /*NX+5 TO COL R LATCH */
      (0179H)     0011100  11  11  0011000  00000000    01    101     1       0    00010  00000000     0       0

 189      /* LOAD ROW UP LATCH     */
 190  189H!                CLR(R7) ;
      (0189H)     1000111  11  00  0011001  00000000    01    000     1       0    00010  00000000     0       0

 191  199H!                LMI(R7) K00005 ;
      (0199H)     0010111  11  00  0011010  00000101    01    000     1       0    00010  00000000     0       0

 192  1A9H!                CMR(R7) ;
      (01A9H)     1110111  11  00  0011011  00000000    01    000     1       0    00010  00000000     0       0

 193  1B9H!                ILR(R6) ;
      (01B9H)     0000110  11  00  0011100  00000000    01    000     1       0    00010  00000000     0       0

 194  1C9H!                ALR(R7) FF1 ;
      (01C9H)     0000111  11  11  0011101  11111111    01    000     1       0    00010  00000000     0       0
```

```
195  1D9H!                LMI(T) FF1 ROT ;  /*NY-5 TO ROW UP LATCH */
     (01D9H)  0011100  11  11  0011110  00000000  01   101   1     0     00010  00000000    0     0

196  /* LOAD ROW DOWN LATCH */
197  1E9H!                LMI(R6) K00005 ;
     (01E9H)  0010110  11  00  0011111  00000101  01   000   1     0     00010  00000000    0     0

198  1F9H!                ILR(R6) ;
     (01F9H)  0000110  11  00  0111000  00000000  01   000   1     0     00010  00000000    0     0

199  1FAH!                LMI(T) FF1 ROT ;  /* NY+5 TO ROW DOWN LATCH */
     (01FAH)  0011100  11  11  0001011  00000000  01   101   1     0     00010  00000000    0     0

200  /* READ BLACK/WHITE CONTROL WORD */
201  0BAH!       COMMON!                      CLR(AC) ;
     (00BAH)  1001101  11  00  0001100  00000000  01   000   1     0     00010  00000000    0     0
```

```
RECORD            CPF    FI  FO    JUMP      KRUSS    PAGEF  CBUS  PAUSEF  STRB  LPCNT    EMIT     MULTIPL  SPRF
NUMBER        6543210    10  10  6543210   76543210    10     210     0      0   43210  76543210     0       0

 202  0C8H!               LMT(AC) K0001D RRM ;
      (00C8H) 0011101    11  00  0001101   00011101    01     110     1      0   00010  00000000     0       0

 203  0D8H!               ACM(AC) ;
      (00D8H) 0001011    11  00  0101110   00000000    01     000     1      0   00010  00000000     0       0

 204  0E8H!               SDR(R5) FF1 ;   /* R/W TO R5  */
      (00E8H) 0100101    11  11  0001111   11111111    01     000     1      0   00010  00000000     0       0

 205  /* READ CURRENT MODE LATCH */
 206  0F8H!               CLR(T) ;
      (00F8H) 1001100    11  00  0010000   00000000    01     000     1      0   00010  00000000     0       0

 207  108H!               LMT(T) K0001C RRM ;
      (0108H) 0011100    11  00  0010001   00011100    01     110     1      0   00010  00000000     0       0

 208  118H!               ACM(AC) ;
      (0118H) 0001011    11  00  0011010   00000000    01     000     1      0   00010  00000000     0       0

 209  1A8H!               SDR(R6) FF1 ; /* MODE TO R6 */
      (01A8H) 0100110    11  11  0010010   11111111    01     000     1      0   00010  00000000     0       0
```

```
210  /* TEST FOR BLACK OR WHITE */
211  128H!              TZR(R5) FF0 ;
     (0128H)  1010101  11  00  0110000  11111111   01    000    1    0    00010  00000000    0    0

212  120H!              CLR(AC) JFL(WHITE,BLACK) ;
     (0120H)  1001101  11  00  1000100  00000000   01    000    1    0    00010  00000000    0    0

213  142H!        WHITE! LMT(AC) K0001B ;
     (0142H)  0011101  11  00  0110001  00011000   01    000    1    0    00010  00000000    0    0

214  141H!              CMR(AC) ;
     (0141H)  1111101  11  00  0010101  00000000   01    000    1    0    00010  00000000    0    0

215  151H!              ANR(R6) ;  /* MODE=MODE.AND.KFFFE7 */
     (0151H)  1000110  11  00  0010110  11111111   01    000    1    0    00010  00000000    0    0

216  161H!              ILR(R6) ;
     (0161H)  0000110  11  00  0010111  00000000   01    000    1    0    00010  00000000    0    0
```

```
XMAS VERS 2.0 CORRELATION MICROCODE ..............................           ERRORS= 0   PAGE 12

RECORD              CPE    FI  FO    JUMP     KRUSS    PAGEF  CBUS  PAUSEF  STRB  LPCNT    EMIT     MULTIPL  SPRF
NUMBER            6543210  10  10  6543210  76543210    10    210     0      0    43210  76543210      0       0

 217  /* WRITE UP-ATED MODE INTO MODE CELL */
 218  171H!               LMI(T) RWM ;
      (0171H)  0011100  11  00  0011000  00000000    01    111     1      0    00010  00000000      0       0

 219  181H!               CLR(T) ;
      (0181H)  1001100  11  00  0110010  00000000    01    000     1      0    00010  00000000      0       0

 220  /* UPDATE MODE LATCH FOR WHITE BOX  */
 221  182H!               LMI(T) K00006 POT PAGE1 JZR(FETCH) ;
      (0182H)  0011100  11  00  0101111  00000110    00    101     1      0    00010  00000000      0       0

 222  143H!         BLACK! LMI(AC) K00010 ;
      (0143H)  0011101  11  00  0110100  00010000    01    000     1      0    00010  00000000      0       0

 223  144H!               OPR(R6) ;
      (0144H)  1100110  11  00  0110111  11111111    01    000     1      0    00010  00000000      0       0

 224  147H!               CLR(AC) ;
      (0147H)  1001101  11  00  0010010  00000000    01    000     1      0    00010  00000000      0       0

 225  127H!               LMI(AC) K00008 ;
      (0127H)  0011101  11  00  0110100  00001000    01    000     1      0    00010  00000000      0       0
```

```
124H!                   CMR(AC) ;
(0124H)  1111101  11  00  0010001  00000000  01  000  1  0  00010  00000000  0  0

114H!                   ANR(R6) ;  /* MODE=MODE.AND.KFFFF7  */
(0114H)  1000110  11  00  0010101  11111111  01  000  1  0  00010  00000000  0  0

154H!                   ILR(R6) ;
(0154H)  0000110  11  00  0010110  00000000  01  000  1  0  00010  00000000  0  0

/* UPDATE MODE FOR BLACK BOX */
164H!                   LMI(T) RWM ;
(0164H)  0011100  11  00  0010111  00000000  01  111  1  0  00010  00000000  0  0

/* UPDATE MODE LATCH FOR BLACK BOX */
174H!                   CLR(T) ;
(0174H)  1001100  11  00  0011000  00000000  01  000  1  0  00010  00000000  0  0

184H!                   LMI(T) K00006 ROT PAGE1 JZR(FETCH) ;
(0184H)  0011100  11  00  0101111  00000110  00  101  1  0  00010  00000000  0  0
```

```
RECORD              CPE   FI  FO    JUMP      KRUSS   PAGEF  CBUS  PAUSEF  STRB  LPCNT     EMIT    MULTIPL  SPRF
NUMBER          6543210   10  10  6543210  76543210    10    210     0      0    43210  76543210     0       0

 234
 235
 236
 237
 238
 239   /* ROUTINE FRO FRM FRM FRM FRO FRM FRM FRM FRM FRM */
 240
 241
 242   078H!       FRM!   CLR(AC) ;
       (0078H)  1001101   11  00  0000001  00000000    01    000     1      0    00010  00000000     0       0

 243   018H!              LMI(AC) K0001C RRM ;
       (0018H)  0011101   11  00  0000010  00011100    01    110     1      0    00010  00000000     0       0

 244   028H!              ACM(AC) ;
       (0028H)  0001011   11  00  0000011  00000000    01    000     1      0    00010  00000000     0       0

 245   038H!              SDR(R6) FF1 ;  /*MODE TO R6 */
       (0038H)  0100110   11  11  0000100  11111111    01    000     1      0    00010  00000000     0       0

 246     /* FORCE REQ BIT HIGH */
 247   048H!       CLR(AC);
       (0048H)  1001101   11  00  0000101  00000000    01    000     1      0    00010  00000000     0       0

 249   058H!              LMI(AC) K00002;
       (0058H)  0011101   11  00  0000110  00000010    01    000     1      0    00010  00000000     0       0
```

```
249  068H!              ORR(R6)  ;  /* MODE=MODE.OR.0002 */
     (0068H)  1100110  11  00  0001000  11111111   01    000    1     0    00010  00000000    0     0

250      /* FORCE R/W BIT LOW */
251  088H!              CLR(AC) ;
     (0088H)  1001101  11  00  0001001  00000000   01    000    1     0    00010  00000000    0     0

252  098H!              LMI(AC) K00001 ;
     (0098H)  0011101  11  00  0001010  00000001   01    000    1     0    00010  00000000    0     0

253  0A8H!              CMR(AC) ;
     (00A8H)  1111101  11  00  0110000  00000000   01    000    1     0    00010  00000000    0     0

254  0A0H!              ANR(R6) ;  /*MODE=MODE.AND.FFFFE */
     (00A0H)  1000110  11  00  0001011  11111111   01    000    1     0    00010  00000000    0     0
```

```
XMAS VERS 2.0 CORRELATION MICROCODE ..............................                     ERRORS= 0   PAGE 14

RECORD                CPE    FI  FO   JUMP     KBUSS    PAGEF  CBUS  PAUSFF   STRB  LPCNT    EMIT    MULTIPL  SPRF
NUMBER            6543210  10  10  6543210  76543210    10    210     0       0    43210  76543210     0       0

  255  080H!                ILR(R6) ;
      (0080H)   0000110  11  00  0001100  00000000    01    000     1       0    00010  00000000     0       0

  256  0C0H!                CLR(T) ;
      (00C0H)   1001100  11  00  0001101  00000000    01    000     1       0    00010  00000000     0       0

  257  0D0H!                LMI(T) K00006 ROT ;
      (00D0H)   0011100  11  00  0001110  00000110    01    101     1       0    00010  00000000     0       0

  258  0E0H!                CLR(T) ;
      (00E0H)   1001100  11  00  0001111  00000000    01    000     1       0    00010  00000000     0       0

  259  0F0H!                LMI(T) K0001C PAGE1 RWM STROBE JZR(FETCH) ;
      (00F0H)   0011100  11  00  0101111  00011100    00    111     1       1    00010  00000000     0       0
```

```
/* ROUTINE STD STD STD STD STD STD STD STD STD*/


    /*  R9=SA    */
/* R8=MEAN*2**5  */
    /*  T=SUM  */
    /*  R7=WORKING  */
    /*  AC=WORKING  */
   /*ROUTINE CALCULATES 4SUM(ABS(I)I-MEAN))  */
   /* I=1,8]  */
073H!       STD!   CLR(T) LCNT1 M10!  /* SUM=0 */
(0073H)  1001100  11  00  0001000  00000000  01    000    1    0    00100  11110110    0    0

   /* CALCULATE STARTING ADDRESS  */
083H!              CLR(R9) !
(0083H)  1001001  11  00  0110101  00000000  01    000    1    0    00010  00000000    0    0

085H!              LMT(R9) K00010 RRM !
(0085H)  0011001  11  00  0001001  00010000  01    110    1    0    00010  00000000    0    0

095H!              ACM(AC) !
(0095H)  0001011  11  00  0001010  00000000  01    000    1    0    00010  00000000    0    0

0A5H!              SDR(R9) FF1 !          /* R9=TP */
(00A5H)  0101001  11  11  0001100  11111111  01    000    1    0    00010  00000000    0    0
```

```
XMAS VERS 2.0 CORRELATION MICROCODE ..............................                          ERRORS= 0   PAGE 15

RECORD                CPE    FI FO   JUMP     KBUSS   PAGEF  CBUS  PAUSFF  STRB  LPCNT   EMIT    MULTIPL  SPRF
NUMBER             6543210   10 10  6543210  76543210   10    210     0      0   43210  76543210     0      0

 281   0C5H!                 LMI(R9) K0001F ;
       (00C5H)     0011001   11 00  0001101  00011111   01    000     1      0   00010  00000000     0      0

 282   005H!                 LMI(R9) K0001F ;   /*R9=SA=IP+62 */
       (0005H)     0011001   11 00  0110111  00011111   01    000     1      0   00010  00000000     0      0

 283   007H!                 LMI(R9) K0000A;   /* R9=SA=IP+72 */
       (0007H)     0011001   11 00  0001100  00001010   01    000     1      0   00010  00000000     0      0

 284   0C7H!                 CLR(R8);
       (00C7H)     1001000   11 00  0110110  00000000   01    000     1      0   00010  00000000     0      0

 285   0C6H!                 LMI(R8) K0001A FE1 RRM;
       (00C6H)     0011000   11 11  0001010  00011010   01    110     1      0   00010  00000000     0      0

 286   0A6H!                 ACM(AC);   /* AC=DX */
       (00A6H)     0001011   11 00  0001001  00000000   01    000     1      0   00010  00000000     0      0

 287   096H!                 ALR(R9);   /*R9=IP+72+DX */
       (0096H)     0001001   11 00  0001000  11111111   01    000     1      0   00010  00000000     0      0

 288   086H!                 LMI(R8) RRM;
       (0086H)     0011000   11 00  0001111  00000000   01    110     1      0   00010  00000000     0      0
```

```
289  0F6H!                ACM(AC);    /* AC=DY */
     (00F6H)  0001011  11  00  0110111  00000000   01   000    1    0   00010  00000000    0    0

290  0F7H!                ADR(R9);
     (00F7H)  0111001  11  00  0010000  11111111   01   000    1    0   00010  00000000    0    0

291  107H!                ALR(AC);    /*AC=DX*2 */
     (0107H)  0001101  11  00  0010001  11111111   01   000    1    0   00010  00000000    0    0

292  117H!                ALR(AC);
     (0117H)  0001101  11  00  0110110  11111111   01   000    1    0   00010  00000000    0    0

293  116H!                ALR(AC);
     (0116H)  0001101  11  00  0010000  11111111   01   000    1    0   00010  00000000    0    0

294  106H!                ALR(AC);    /* AC=DX*16 */
     (0106H)  0001101  11  00  0001110  11111111   01   000    1    0   00010  00000000    0    0
```

```
RECORD              CPE      FI  FO    JUMP     KBUSS    PAGEF  CBUS  PAUSEF  STRB  LPCNT     EMIT     MULTIPL  SPRF
NUMBER            6543210    10  10  6543210  76543210    10    210     0      0    43210   76543210     0       0

 295  0F6H!                  ADR(R9):   /* R9=IP+72+DX+17*D;  */
      (00F6H)   0111001      11  00  0110101  11111111    01    000     1      0    00010   00000000     0       0

 296       /*  FETCH    MEAN  VALUE  */
 297  0F5H!                  CLR(AC) LCNT2 M10;
      (00F5H)   1001101      11  00  0010000  00000000    01    000     1      0    01000   11110110     0       0

 298  105H!                  LMI(AC) K00014 RRM ;
      (0105H)   0011101      11  00  0010001  00010100    01    110     1      0    00010   00000000     0       0

 299  115H!                  ACM(AC) ;
      (0115H)   0001011      11  00  0010010  00000000    01    000     1      0    00010   00000000     0       0

 300  125H!                  CMP(AC) ;
      (0125H)   1111101      11  00  0110110  00000000    01    000     1      0    00010   00000000     0       0

 301  126H!                  SDR(R8) FFI ;     /* R8=COMP OF MEANS */
      (0126H)   0101000      11  11  0010011  11111111    01    000     1      0    00010   00000000     0       0

 302       /* BEGIN  LOOPS  */
 303  136H!                  ILR(R8) ;
      (0136H)   0001000      11  00  0110111  00000000    01    000     1      0    00010   00000000     0       0
```

```
304  137H!               SOR(R7) FF1 ; /*MEAN TO R7 */
     (0137H)  0100111  11  11  0110010  11111111  01   000   1   0   00010  00000000   0   0

305  132H!      LXXP!  LMT(R9) FF1 RRM ;
     (0132H)  0011001  11  11  0010101  00000000  01   110   1   0   00010  00000000   0   0

306  152H!      LXXP2! ACM(AC);
     (0152H)  0001011  11  00  0110000  00000000  01   000   1   0   00010  00000000   0   0

307  150H!               ALR(AC) ;
     (0150H)  0001101  11  00  0010100  11111111  01   000   1   0   00010  00000000   0   0

308  140H!               ALR(AC) ;
     (0140H)  0001101  11  00  0010111  11111111  01   000   1   0   00010  00000000   0   0

309  170H!               ALR(AC) ;
     (0170H)  0001101  11  00  0011000  11111111  01   000   1   0   00010  00000000   0   0
```

```
XMAS VERS 2.0 CORRELATION MICROCODE ..............................          ERRORS= 0   PAGE 17

RECORD              CPE     FI  FO   JUMP     KBUSS    PAGEF  CBUS  PAUSEF  STRB  LPCNT    EMIT     MULTIPL  SPRF
NUMBER            6543210   10  10  6543210  76543210   10    210     0      0    43210  76543210     0       0

 310  189H!                 ALR(AC) ;
      (0189H)     0001101   11  00  0011001  11111111   01    000     1      0    00010  00000000     0       0

 311  190H!                 ALR(AC) ;
      (0190H)     0001101   11  00  0110010  11111111   01    000     1      0    00010  00000000     0       0

 312  192H!                 ALR(R7) FF1;  /* (2**5*C(SA)-MEANS)  TO AC */
      (0192H)     0000111   11  11  0110110  11111111   01    000     1      0    00010  00000000     0       0

 313  196H!                 NOP(AC);
      (0196H)     1101101   11  00  0010101  00000000   01    000     1      0    00010  00000000     0       0

 314     /* TAKE ABSOLUTE  VALUE    */
 315  156H!                 TZR(AC) K80000 INH ;
      (0156H)     1011101   11  00  0010110  10000000   01    001     1      0    00010  00000000     0       0

 316  166H!                 NOP(AC) JFL(APX.ANX) ;
      (0166H)     1101101   11  00  1000110  00000000   01    000     1      0    00010  00000000     0       0

 317  163H!         ANX!    CIA(AC) FF1 ;
      (0163H)     0011111   11  11  0110010  00000000   01    000     1      0    00010  00000000     0       0

 318  162H!         APX!    ALR(T) FF0        ;  /* SUM=SUM+ABS(C(A(I))-MEAN) */
      (0162H)     0001100   11  00  0111000  11111111   01    000     1      0    00010  00000000     0       0
```

```
319  168H!                ILR(R8) ITCNT1:
     (0168H)  0001000  11  00  0110000  00000000  01   000   1    0   00111  00000000   0    0

320  160H!                SDR(R7) FF1 JFL(LXXP.CONTXX):
     (0160H)  0100111  11  11  1000011  11111111  01   000   1    0   00010  00000000   0    0

321  133H!          ONTXX! LMI(R9) K00008 ITCNT2:
     (0133H)  0011001  11  00  0110100  00001000  01   000   1    0   01011  00000000   0    0

322  134H!                LMI(R9) FF1 RRM LCNT1 M10 JFL(LXXP2.CONTX2):
     (0134H)  0011001  11  11  1000101  00000000  01   110   1    0   00100  11110110   0    0

323  153H!          CONTX2!                    ILR(T) :
     (0153H)  0001100  11  00  0110111  00000000  01   000   1    0   00010  00000000   0    0

324  157H!                ALR(AC) :
```

```
RECORD              PE      FI  FO   JUMP     KRUSS    PAGEF  CBUS  PAUSEF  STRB  LPCNT   EMIT     MULTIPL  SPRF
NUMBER            6543210   10  10  6543210  76543210   10    210     0      0    43210  76543210    0       0

        (0157H)   0001101   11  00  0010110  11111111   01    000     1      0    00010  00000000    0       0

  325  167H!                ALR(AC) ; /*SUM=4(SUM) */
        (0167H)   0001101   11  00  0010111  11111111   01    000     1      0    00010  00000000    0       0

  326     /*   ST-PE  STD  IN    MEMORY    */
  327  177H!                CLR(R9) ;
        (0177H)   1001001   11  00  0011000  00000000   01    000     1      0    00010  00000000    0       0

  328  187H!                LMI(R9) K00015 RWM  PAGE1 JZR(FETCH) ;
        (0187H)   0011001   11  00  0101111  00010101   00    111     1      0    00010  00000000    0       0

  329
  330
  331
  332
  333
  334 /* ROUTINE REF REF REF REF REF REF REF REF REF REF */
  335
  336
  337     /* R8=STA   */
  338     /* R9=SA     */
  339      /*  FORM STARTING ADDRESS SA    */
  340  07DH!       REF!     CLR(AC) ;
        (007DH)   1001101   11  00  0010011  00000000   01    000     1      0    00010  00000000    0       0
```

```
341  13DH!                 LMI(AC) K00010 RRM ;
     (013DH)  0011101  11  00  0110101  00010000   01   110   1   0   00010  00000000   0   0

342  135H!                 ACM(AC) ;  /*IP TO T   */
     (0135H)  0001011  11  00  0010100  00000000   01   000   1   0   00010  00000000   0   0

343  145H!                 LMI(AC) K0001F ;
     (0145H)  0011101  11  00  0010101  00011111   01   000   1   0   00010  00000000   0   0

344  155H!                 LMI(AC) K0001F ;
     (0155H)  0011101  11  00  0010110  00011111   01   000   1   0   00010  00000000   0   0

345  165H!                 LMI(AC) K0000A ;
     (0165H)  0011101  11  00  0010111  00001010   01   000   1   0   00010  00000000   0   0

346  175H!                 SDR(R9) FF1 ;  /*R9=IP+72=SA */
     (0175H)  0101001  11  11  0110110  11111111   01   000   1   0   00010  00000000   0   0
```

```
XMAS VERS 2.0 CORRELATION MICROCODE ..............................              ERRORS= 0    PAGE 19

RECORD                CPE    FI  FO    JUMP     KRUSS   PAGEF  CBUS  PAUSEF  STRB  LPCNT    EMIT     MULTIPL  SPRF
NUMBER            6543210    10  10  6543210  76543210    10    210     0      0   43210  76543210      0       0

 347   176H!                 CLR(R8):
       (0176H)    1001000    11  00  0011011  00000000    01    000     1      0   00010  00000000      0       0

 348   1B6H!                 LMI(R8) RRM FF1 <0001A:
       (01B6H)    0011000    11  11  0011100  00011010    01    110     1      0   00010  00000000      0       0

 349   1C6H!                 ACM(AC): /*AC=DX*/
       (01C6H)    0001011    11  00  0011101  00000000    01    000     1      0   00010  00000000      0       0

 350   1D6H!                 ADR(R9) :  /*R9=IP+SA+DX*/
       (01D6H)    0111001    11  00  0011110  11111111    01    000     1      0   00010  00000000      0       0

 351   1E6H!                 LMI(R8) RPM:
       (01E6H)    0011000    11  00  0110111  00000000    01    110     1      0   00010  00000000      0       0

 352   1E7H!                 ACM(AC):  /*AC=DY*/
       (01E7H)    0001011    11  00  0011101  00000000    01    000     1      0   00010  00000000      0       0

 353   1D7H!                 ADR(R9):  /*R9=IP+DX+72+DY*/
       (01D7H)    0111001    11  00  0011100  11111111    01    000     1      0   00010  00000000      0       0

 354   1C7H!                 ALR(AC):  /*AC=DY*2 */
       (01C7H)    0001101    11  00  0011011  11111111    01    000     1      0   00010  00000000      0       0
```

```
355  1B7H!              ALR(AC);   /*AC=DY*4 */
     (01B7H)  0001101  11  00  0011010  11111111   01   000   1   0   00010  00000000   0   0

356  1A7H!              ALR(AC);    /*AC=DY*8 */
     (01A7H)  0001101  11  00  0110110  11111111   01   000   1   0   00010  00000000   0   0

357  1A6H!              ALR(AC);   /*AC=DY*16  */
     (01A6H)  0001101  11  00  0011000  11111111   01   000   1   0   00010  00000000   0   0

358  186H!              ADR(R9);   /*R9=IP+72+DX+DY*17 */
     (0186H)  0111001  11  00  0110101  11111111   01   000   1   0   00010  00000000   0   0

359      /* FORM  STORAGE ADDRESS STA      */
360  185H!              CLR(AC) LCNT1 M10;
     (0185H)  1001101  11  00  0011001  00000000   01   000   1   0   00100  11110110   0   0

361  195H!              LMI(AC) K00011 R3M ;
```

```
XMAS VERS 2.0 CORRELATION MICROCODE .............................                  ERRORS= 0   PAGE 20

RECORD                CPF      FI  FO   JUMP     KBUSS    PAGEF  CBUS  PAUSFF  STRB  LPCNT  EMIT      MULTIPL  SPRF
NUMBER                6543210  10  10  6543210  76543210   10    210     0      0    43210  76543210    0       0

           (0195H)  0011101  11  00  0011010  00010001   01    110     1      0    00010  00000000    0       0

  362  1A5H!                   ACM(AC) LCNT2 M10;
       (01A5H)  0001011  11  00  0011011  00000000   01    000     1      0    01000  11110110    0       0

  363  1B5H!                   SDR(R8) FF1 ;  /*R8=RP=STA    */
       (01B5H)  0101000  11  11  0110100  11111111   01    000     1      0    00010  00000000    0       0

  364  1B4H!                   CLR(T);
       (01B4H)  1001100  11  00  0110010  00000000   01    000     1      0    00010  00000000    0       0

  365.      /* START LOOPS     */
  366  1B2H!       LX1!    LMI(R9) FF1 RRM ;
       (01B2H)  0011001  11  11  0011100  00000000   01    110     1      0    00010  00000000    0       0

  367  1C2H!       LX2!    ACM(AC);
       (01C2H)  0001011  11  00  0110100  00000000   01    000     1      0    00010  00000000    0       0

  368  1C4H!                   LMI(R8) FF1 RWM JFL(LX1,CX1) ;
       (01C4H)  0011000  11  11  1001011  00000000   01    111     1      0    00010  00000000    0       0

  369  1B3H!       CX1!    LMI(R9) K00008 ITCNT2 ;
       (01B3H)  0011001  11  00  0110001  00001000   01    000     1      0    01011  00000000    0       0
```

```
181H!                LMI(R9) FFI RRM LCNT1 M10 JFL(LX2.CX2);
(0181H)   0011001   11  11  1001100  00000000  01  110   1   0   00100  11110110   0   0

1C3H!        CX2!    NOP(AC) PAGE1 JZR(FETCH);
(01C3H)   1101101   11  00  0101111  00000000  00  000   1   0   00010  00000000   0   0





/* ROUTINE WAT WAT WAT WAT WAT WAT WAT WAT WAT WAT */


079H!        WAT!    NOP(AC) RIN ;
(0079H)   1101101   11  00  0000001  00000000  01  100   1   0   00010  00000000   0   0

019H!                LDI(AC) FF1 ;
(0019H)   0101111   11  11  0000010  11111111  01  000   1   0   00010  00000000   0   0
```

```
XMAS VERS 2.0 CORRELATION MICROCODE ...............................                    ERRORS= 0   PAGE 21

RECORD                CPE      FI  FO    JUMP      KBUSS     PAGEF  CBUS  PAUSEF  STRB  LPCNT    EMIT     MULTIPL  SPRF
NUMBER              6543210   10  10  6543210  76543210    10    210     0       0   43210  76543210     0       0

 382  029H!                   TZR(AC) K80000 ;
      (0029H)       1011101   11  00  0000011  10000000    01    000     1       0   00010  00000000     0       0

 383  039H!                   NOP(AC) JFL(NRDY.RDY) ;
      (0039H)       1101101   11  00  1000011  00000000    01    000     1       0   00010  00000000     0       0

 384  03BH!          RDY!     NOP(AC) JZR(FETCH) PAGE1 ;
      (003BH)       1101101   11  00  0101111  00000000    00    000     1       0   00010  00000000     0       0

 385  03AH!         NRDY!     NOP(AC) JZR(XX2) ;
      (003AH)       1101101   11  00  0101001  00000000    01    000     1       0   00010  00000000     0       0

 386  009H!           X2!     NOP(AC) JCC(WAT) ;
      (0009H)       1101101   11  00  0000111  00000000    01    000     1       0   00010  00000000     0       0
```

```
/* ROUTINE MNA MNV MNV MNV MNA MNV MNV MNV MNV  */


      /* R8=WORKING */
   /* R9=SA    */
   /* T=SUM    */
   /* AC=WORKING   */
   /*ROUTINE TO CALCULATE MEAN OF CENTER 9X9 IMG  */
07CH!       MNV!   CLR(T) LCNT1 M10;
 (007CH)  1001100  11  00  0000001  00000000  01   000   1   0   00100  11110110   0   0

   /*  CALCULATE   STARTING  ADDRESS  */
01CH!              CLR(R9) LCNT2 M10;
 (001CH)  1001001  11  00  0110100  00000000  01   000   1   0   01000  11110110   0   0

014H!              LMI(R9) K00010 RRM ;
 (0014H)  0011001  11  00  0000010  00010000  01   010   1   0   00010  00000000   0   0

024H!              ACM(AC) ;
 (0024H)  0001011  11  00  0000000  00000000  01   000   1   0   00010  00000000   0   0

004H!              SDR(R9) FF1;   /* R9=IP */
 (0004H)  0101001  11  11  0000011  11111111  01   000   1   0   00010  00000000   0   0
```

```
XMAS VERS 2.0 CORRELATION MICROCODE ..............................                ERRORS= 0   PAGE 22

RECORD            CPE      FI  FO   JUMP     KBUSS    PAGEF  CBUS  PAUSFF  STRB  LPCNT    EMIT     MULTIPL  SPRF
NUMBER          6543210   10  10  6543210  76543210    10    210     0      0    43210  76543210     0       0

 406  034H!                CLR(R8);
      (0034H)   1001000   11  00  0110110  00000000    01    000     1      0    00010  00000000     0       0

 407  036H!               LMT(R8) K0001A FF1 RRM;
      (0036H)   0011000   11  11  0000010  00011010    01    110     1      0    00010  00000000     0       0

 408  026H!                ACM(AC);   /* AC=DX */
      (0026H)   0001011   11  00  0110101  00000000    01    000     1      0    00010  00000000     0       0

 409  025H!                LMT(AC) K0001F;
      (0025H)   0011101   11  00  0000011  00011111    01    000     1      0    00010  00000000     0       0

 410  035H!                LMT(AC) K0001F;   /* AC=DX+62 */
      (0035H)   0011101   11  00  0110111  00011111    01    000     1      0    00010  00000000     0       0

 411  037H!                LMT(AC) K0000A;   /* AC=DX+72 */
      (0037H)   0011101   11  00  0000001  00001010    01    000     1      0    00010  00000000     0       0

 412  017H!                ALR(R9);   /*R9,AC=IP+DX+72 */
      (0017H)   0001001   11  00  0000010  11111111    01    000     1      0    00010  00000000     0       0

 413  027H!                LMT(R8)  RRM;
      (0027H)   0011000   11  00  0000100  00000000    01    110     1      0    00010  00000000     0       0
```

```
414  047H!                ACM(AC);  /* AC=DY */
     (0047H)  0001011  11  00  0000110  00000000  01  000  1  0  00010  00000000  0  0

415  067H!                ADR(R9);  /*R9=IP+DX+72+DY  */
     (0067H)  0111001  11  00  0001000  11111111  01  000  1  0  00010  00000000  0  0

416  087H!                ALR(AC);  /* AC=DX*2 */
     (0087H)  0001101  11  00  0001010  11111111  01  000  1  0  00010  00000000  0  0

417  047H!                ALR(AC);  /* AC=DX*4 */
     (0047H)  0001101  11  00  0001011  11111111  01  000  1  0  00010  00000000  0  0

418  087H!                ALR(AC);  /* AC=DX*8  */
     (0087H)  0001101  11  00  0000101  11111111  01  000  1  0  00010  00000000  0  0

419  057H!                ALR(AC);  /* AC=DX*16   */
     (0057H)  0001101  11  00  0110100  11111111  01  000  1  0  00010  00000000  0  0
```

```
RECORD                CPF     FI  FO  JUMP     KBUSS     PAGEF  CBUS  PAUSFF  STRB  LPCNT  EMIT      MULTIPL  SPRF
NUMBER                6543210 10  10  6543210  76543210  10     210   0       0     43210  76543210  0        0

 420  054H!                   ADR(R9);  /* R9=SA=IP+72+DX+DY*17  */
      (0054H)         0111001 11  00  0110001  11111111  01     000   1       0     00010  00000000  0        0

 421  051H!                   CLR(AC);
      (0051H)         1001101 11  00  0110010  00000000  01     000   1       0     00010  00000000  0        0

 422  052H!    LOOP1! LMI(R9) FF1 RRM ITCNT1 ;  /*SA=SA+1 */
      (0052H)         0011001 11  11  0000100  00000000  01     110   1       0     00111  00000000  0        0

 423  042H!     LXXX! AMA(AC) JFL(LOOP1,CONT1);
      (0042H)         0001011 11  00  1000101  11111111  01     000   1       0     00010  00000000  0        0

 424  053H!     CONT1! LMI(R9) K00008 ITCNT2 ;
      (0053H)         0011001 11  00  0110101  00001000  01     000   1       0     01011  00000000  0        0

 425  055H!           LMI(R9) FF1     LCNT1 M10 JFL(LOOP2,CONT2);
      (0055H)         0011001 11  11  1000110  00000000  01     000   1       0     00100  11110110  0        0

 426  062H!    LOOP2! NOP(AC) ITCNT1 RRM JCC(LXXX);
      (0062H)         1101101 11  00  0000100  00000000  01     110   1       0     00111  00000000  0        0

 427  063H!    CONT2! CLR(R9) ;
      (0063H)         1001001 11  00  0110100  00000000  01     000   1       0     00010  00000000  0        0
```

```
064H!                LMI(R9) K00014 ;          /* ADR OF MEANS */
(0064H)  0011001  11  00  0110101  00010100  01   000   1    0   00010  00000000   0   0

065H!                NOP(AC) RWM PAGE1 JZR(FETCH);
(0065H)  1101101  11  00  0101111  00000000  00   111   1    0   00010  00000000   0   0





/*ROUTINE CROSS CROSS CROSS CROSS CROSS CROSS  */


074H!       CROSS! CLR(AC) ;
(0074H)  1001101  11  00  0001000  00000000  01   000   1    0   00010  00000000   0   0

084H!                LMI(AC) K00018 RRM ;
(0084H)  0011101  11  00  0001001  00011000  01   110   1    0   00010  00000000   0   0
```

```
XMAS VERS 2.0 CORRELATION MICROCODE ..............................          ERRORS= 0   PAGE 24

RECORD                 CPE     FI  FO   JUMP      KBUSS    PAGEF  CBUS  PAUSEF  STRB  LPCNT    EMIT     MULTIPL  SPRF
NUMBER               6543210  10  10  6543210  76543210    10    210     0      0   43210  76543210     0       0

 440  094H!                   ACM(AC) ;
      (0094H)        0001011  11  00  0001010  00000000    01    000     1      0   00010  00000000     0       0

 441  0A4H!                   CLR(T) ;
      (00A4H)        1001100  11  00  0001011  00000000    01    000     1      0   00010  00000000     0       0

 442  0B4H!                   LMI(T) FE1 ROT ;  /* D0=NX COL LL */
      (00B4H)        0011100  11  11  0001100  00000000    01    101     1      0   00010  00000000     0       0

 443  0C4H!                   LMI(T) FE1 ROT ;  /* D1=NX COL PL */
      (00C4H)        0011100  11  11  0001101  00000000    01    101     1      0   00010  00000000     0       0

 444  0D4H!                   CLR(AC) ;
      (00D4H)        1001101  11  00  0001110  00000000    01    000     1      0   00010  00000000     0       0

 445  0E4H!                   LMI(AC) K00019 R2M ;
      (00E4H)        0011101  11  00  0001111  00011001    01    110     1      0   00010  00000000     0       0

 446  0F4H!                   ACM(AC) ;
      (00F4H)        0001011  11  00  0110101  00000000    01    000     1      0   00010  00000000     0       0

 447  0F5H!                   LMI(T) FE1 ROT ;  /*D2=NY ROW UL  */
      (00F5H)        0011100  11  11  0001011  00000000    01    101     1      0   00010  00000000     0       0
```

```
448  0B5H!               LMI(T) FF1 ROT ;  /* D3=NY ROW DL  */
     (00B5H)  0011100  11  11  0011111  00000000   01    101     1     0    00010  00000000    0     0

449      /* READ BLACK/WHITE CONTROL WORD  */
450  1F5H!               CLR(AC) ;
     (01F5H)  1001101  11  00  0111100  00000000   01    000     1     0    00010  00000000    0     0

451  1FCH!               LMI(AC) K0001D R3M ;
     (01FCH)  0011101  11  00  0011110  00011101   01    110     1     0    00010  00000000    0     0

452  1ECH!               ACM(AC) ;
     (01ECH)  0001011  11  00  0111101  00000000   01    000     1     0    00010  00000000    0     0

453  1EDH!               SDR(R5) FF1 ;  /*B/W TO R5 */
     (01EDH)  0100101  11  11  0010111  11111111   01    000     1     0    00010  00000000    0     0

454      /* READ CURRENT  MODE  */
```

```
RECORD              CPF     FI  FO  JUMP     KRUSS     PAGEF  CBUS  PAUSFF  STRB  LPCNT  EMIT      MULTIPL  SPRF
NUMBER              6543210 10  10  6543210  76543210  10     210   0       0     43210  76543210  0        0

 455  170H!                 CLR(T) ;
      (0170H)       1001100 11  00  0011000  00000000  01     000   1       0     00010  00000000  0        0

 456  180H!                 LMI(T) K0001C RRM ;
      (0180H)       0011100 11  00  0011001  00011100  01     110   1       0     00010  00000000  0        0

 457  190H!                 ACM(AC) ;
      (0190H)       0001011 11  00  0011010  00000000  01     000   1       0     00010  00000000  0        0

 458  1A0H!                 SDR(R6) FF1 ;  /* MODE TO R6 */
      (01A0H)       0100110 11  11  0011011  11111111  01     000   1       0     00010  00000000  0        0

 459        /*  TEST FOR BLACK OR WHITE  */
 460  1B0H!                 TZR(R5) FF0 ;
      (01B0H)       1010101 11  00  0011100  11111111  01     000   1       0     00010  00000000  0        0

 461  1C0H!                 CLR(AC) JFL(WHITE2,BLACK2) ;
      (01C0H)       1001101 11  00  1001101  00000000  01     000   1       0     00010  00000000  0        0

 462  1DAH!        WHITE2!                     LMI(AC) K00010 ;
      (01DAH)       0011101 11  00  0111110  00010000  01     000   1       0     00010  00000000  0        0

 463  1DEH!                 CMP(AC) ;
      (01DEH)       1111101 11  00  0010101  00000000  01     000   1       0     00010  00000000  0        0
```

```
464  15EH!                  ANR(R6) ;
     (015EH)  1000110  11  00  0010110  11111111  01  000  1  0  00010  00000000  0  0

465  16EH!                  CLR(AC) ;
     (016EH)  1001101  11  00  0010111  00000000  01  000  1  0  00010  00000000  0  0

466  17EH!                  LMI(AC) K00008 ;
     (017EH)  0011101  11  00  0010100  00001000  01  000  1  0  00010  00000000  0  0

467  14EH!                  ORR(R6);
     (014EH)  1100110  11  00  0011000  11111111  01  000  1  0  00010  00000000  0  0

468  18EH!                  ILR(R6) ;
     (018EH)  0000110  11  00  0011001  00000000  01  000  1  0  00010  00000000  0  0

469  19EH!                  LMI(T) RWM ;
```

```
XMAS VERS 2.0 CORRELATION MICROCODE ..............................                ERRORS= 0   PAGE 26

RECORD            CPE     FI  FO   JUMP     KBUSS   PAGEF  CBUS  PAUSEF  STRB  LPCNT    EMIT    MULTIPL  SPRF
NUMBER          6543210   10  10 6543210  76543210   10    210     0      0    43210  76543210     0       0

       (019EH)  0011100   11  00 0011010  00000000   01    111     1      0    00010  00000000     0       0

  470  1AFH!              CLR(T) :
       (01AFH)  1001100   11  00 0011011  00000000   01    000     1      0    00010  00000000     0       0

  471  1BFH!              LMI(T) K00006 ROT PAGE1 JZR(FETCH) :
       (01BFH)  0011100   11  00 0101111  00000110   00    101     1      0    00010  00000000     0       0

  472  1DBH!        BLACK2!                   LMI(AC) K00012 :
       (01DBH)  0011101   11  00 0111101  00011000   01    000     1      0    00010  00000000     0       0

  473  1DDH!              ORR(R6) :
       (01DDH)  1100110   11  00 0111111  11111111   01    000     1      0    00010  00000000     0       0

  474  1DFH!              ILR(R6) :
       (01DFH)  0000110   11  00 0011011  00000000   01    000     1      0    00010  00000000     0       0

  475  1BFH!              LMI(T) RWM :
       (01BFH)  0011100   11  00 0011100  00000000   01    111     1      0    00010  00000000     0       0
```

```
476  1CEH!            CLR(T) :
     (01CEH)  1001100  11  00  0111110  00000000  01   000   1   0   00010  00000000   0   0

477  1CEH!            LMI(T) K00006 ROT PAGE1 JZR(FETCH) :
     (01CEH)  0011100  11  00  0101111  00000110  00   101   1   0   00010  00000000   0   0

478  00FH!      FETCH! NOP(AC) JZR(FETCH) :
     (000FH)  1101101  11  00  0101111  00000000  01   000   1   0   00010  00000000   0   0

479  EOF

NO PROGRAM ERRORS
END OF PROGRAM
```

XMAS VERS 2.0 CORRELATION MICROCODE .... ERRORS= 0 PAGE 27

MICROPROGRAM MEMORY IMAGE

	0H	1H	2H	3H	4H	5H	6H	7H	8H	9H	AH	BH	CH	DH	EH	FH
0000H					JCC 0034H 405 1					JCC 0079H 386 1						JZR 000FH 478 17
0010H					JCC 0024H 403 1			JCC 0027H 412 1	JCC 0028H 243 1	JCC 0029H 381 1			JCR 0014H 402 1	JCC 002DH 128 1	JCC 002EH 123 1	JCC 002FH 115 1
0020H					JCC 0004H 404 1	JCC 0035H 409 1	JCR 0025H 408 1	JCC 0047H 413 1	JCC 0038H 244 1	JCC 0039H 382 1				JCC 004DH 131 1	JCC 003EH 125 1	JCC 003FH 116 1
0030H					JCR 0036H 406 1	JCR 0037H 410 1	JCC 0026H 407 1	JCC 0017H 411 1	JCC 0048H 245 1	JFL 003AH 383 1	JZR 0009H 385 1	JZR 000FH 384 1		JCC 001DH 127 1	JCR 003DH 126 1	JCC 004FH 117 1
0040H			JFL 0052H 423 2					JCC 0067H 414 1	JCC 0058H 247 1					JCC 005DH 132 1		JCC 005FH 118 1
0050H		JCR 0052H 421 1	JCC 0042H 422 2	JCR 0055H 424 1	JCR 0051H 420 1	JFL 0062H 425 1		JCR 0054H 419 1	JCC 0068H 248 1					JCC 006DH 133 1	JPR 0070H 56 0	JCC 006FH 120 1
0060H			JCC 0042H 426 1	JCR 0056H 427 1	JCR 0065H 428 1	JZR 000FH 429 1		JCC 0087H 415 1	JCC 0088H 249 1					JCC 01FDH 134 1	JCC 001EH 122 1	JCR 006FH 121 1
0070H	JZR 003FH 58 1	JZR 000FH 59 1	JZR 000FH 60 1	JCC 0083H 275 1	JCC 0084H 434 1	JZR 000FH 61 1	JCR 0076H 79 2	JCC 0097H 170 1	JCC 0018H 242 1	JCC 0019H 380 2	JCC 008AH 94 1	JZR 000FH 71 1	JCC 001CH 400 1	JCC 0130H 340 1	JCR 007EH 80 2	JZR 000FH 81 1

XMAS VERS 2.0 CORRELATION MICROCODE .... ERRORS= 0 PAGE 28

MICROPROGRAM MEMORY IMAGE

	0H	1H	2H	3H	4H	5H	6H	7H	8H	9H	AH	BH	CH	DH	EH	FH
008H				JCR 0085H 277 1	JCC 0094H 439 1	JCC 0095H 278 1	JCC 00F6H 288 1	JCC 00A7H 416 1	JCC 0098H 251 1		JCR 00BFH 95 1			JCC 009DH 136 1		JCC 009FH 96 1
009H					JCC 00A4H 440 1	JCC 00A5H 279 1	JCC 00B6H 287 1	JCP 0099H 171 1	JCC 00A8H 252 1	JCC 00A9H 172 1				JCC 00EDH 137 1		JCC 00AFH 97 1
00AH	JCC 00B0H 254 1				JCC 00B4H 441 1	JCC 00C5H 280 1	JCC 0096H 286 1	JCC 00B7H 417 1	JCR 00B8H 253 1	JCC 00B9H 173 1						JCC 00BFH 98 1
00BH	JCC 00C0H 255 1				JCC 00C4H 442 1	JCC 01F5H 448 1		JCC 0057H 418 1	JCC 00C8H 261 1	JCC 00C9H 174 1						JCC 00CFH 99 1
00CH	JCC 00D0H 256 1				JCC 00D4H 443 1	JCC 00D5H 281 1	JCC 00A6H 285 1	JCP 00C6H 284 1	JCC 00D8H 262 1	JCC 00D9H 175 1						JCC 00DFH 100 1
00DH	JCC 00E0H 257 1				JCC 00E4H 444 1	JCR 00D7H 282 1		JCC 00C7H 283 1	JCC 00E8H 263 1	JCC 00E9H 176 1						JCC 00EFH 101 1
00EH	JCC 00F0H 258 1				JCC 00F4H 445 1	JCC 0105H 297 1	JCR 00E5H 295 1		JCC 00F8H 264 1	JCC 00F9H 178 1				JCC 00FDH 138 1		JCC 00FFH 102 1
00FH	JZR 000FH 259 1				JCR 00F5H 446 1	JCC 0085H 447 1	JCR 00F7H 289 1	JCC 0107H 290 1	JCC 0108H 266 1	JCC 0109H 179 1				JCC 011DH 139 1		JCC 010FH 103 1

XMAS VERS 2.0 CORRELATION MICROCODE .... ERRORS= 0 PAGE 29

MICROPROGRAM MEMORY IMAGE

	0H	1H	2H	3H	4H	5H	6H	7H	8H	9H	AH	BH	CH	DH	EH	FH
010H						JCC 0115H 298 1	JCC 00E6H 294 1	JCC 0117H 291 1	JCC 0119H 207 1	JCC 0119H 180 1						JCC 011FH 104 1
011H					JCC 0154H 227 1	JCC 0125H 299 1	JCC 0106H 293 1	JCR 0116H 292 1	JCC 01A9H 208 1	JCC 0129H 181 1				JCC 012DH 140 1		JCC 012FH 105 1
012H	JFL 0142H 212 1				JCC 0114H 225 1	JCR 0126H 300 1	JCC 0136H 301 1	JCR 0124H 225 1	JCR 0120H 211 1	JCC 0139H 182 1	JCC 013AH 143 2	JCR 012CH 146 1	JFL 015AH 147 1	JCR 012AH 141 1		JCC 013FH 105 1
013H			JCC 0152H 305 2	JCR 0134H 321 1	JFL 0152H 322 1	JCC 0145H 342 1	JCR 0137H 303 1	JCR 0132H 304 1		JCC 0149H 183 1	JCR 013CH 144 2		JFL 012AH 145 1	JCR 0135H 341 1		JCC 014FH 107 1
014H	JCC 017[illegible]H 3[illegible]9 1	JCC 0151H 214 1	JCR 0141H 213 1	JCR 0144H 222 1	JCR 0147H 223 1	JCC 0155H 343 1		JCC 0127H 224 1		JCC 0159H 184 1					JCC 018FH 467 1	JCC 015FH 109 1
015H	JCC 014[illegible]H 3[illegible]7 1	JCC 0161H 215 1	JCR 0150H 306 2	JCR 0157H 323 1	JCC 0164H 228 1	JCC 0165H 344 1	JCC 0166H 315 1	JCC 0167H 324 1		JCC 0169H 186 1	JCC 013AH 148 1	JCR 015CH 150 1	JCC 016CH 151 1		JCC 016FH 464 1	JCC 016FH 109 1
016H	JFL 0132H 320 1	JCC 0171H 216 1	JCR 0168H 318 2	JCR 0162H 317 1	JCC 0174H 230 1	JCC 0175H 345 1	JFL 0162H 316 1	JCC 0177H 325 1	JCR 0160H 319 1	JCC 0179H 187 1			JCC 017CH 152 1		JCC 017EH 465 1	JCC 017FH 110 1
017H	JCC 0191H 3[illegible]9 1	JCC 0181H 218 1			JCC 0184H 232 1	JCR 0176H 346 1	JCC 0186H 347 1	JCC 0187H 327 1		JCC 0189H 189 1			JCC 018CH 153 1	JCC 018DH 455 1	JCC 014EH 466 1	JCC 018FH 111 1

XMAS VERS 2.0 CORRELATION MICROCODE .... ERRORS= 0 PAGE 30

MICROPROGRAM MEMORY IMAGE

	0H	1H	2H	3H	4H	5H	6H	7H	8H	9H	AH	BH	CH	DH	EH	FH
018H	JCC 0190H 310 1	JCR 0182H 219 1	JZR 000FH 221 1		JZR 000FH 233 1	JCC 0195H 360 1	JCR 0185H 358 1	JZR 000FH 328 1		JCC 0199H 190 1			JCC 019CH 154 1	JCC 019DH 456 1	JCC 019EH 468 1	JCC 019FH 112 1
019H	JCR 0192H 311 1		JCR 0196H 312 1			JCC 0145H 361 1	JCC 0156H 313 1			JCC 01A9H 191 1			JCC 01ACH 155 1	JCC 01ADH 457 1	JCC 01AEH 469 1	JCC 01AFH 113 1
01AH						JCC 01B5H 362 1	JCC 01B6H 357 1	JCR 0146H 356 1	JCC 012BH 269 1	JCC 01B9H 192 1			JCC 01BCH 156 1	JCC 01BDH 458 1	JCC 01BEH 470 1	JCC 001FH 114 1
01BH		JFL 01C2H 370 1	JCC 01C2H 366 2	JCR 01B1H 369 1	JCR 01B2H 364 1	JCR 01B4H 363 1	JCC 01C6H 348 1	JCC 0147H 355 1		JCC 01C9H 193 1			JCC 01CCH 157 1	JCC 01CDH 460 1	JZR 000FH 471 1	JCC 01CFH 475 1
01CH			JCR 01C4H 367 2	JZR 000FH 371 1	JFL 01B2H 368 1		JCC 01D6H 349 1	JCC 01B7H 354 1		JCC 01D9H 194 1			JCC 01DCH 158 1	JFL 01DAH 461 1	JZR 000FH 477 1	JCR 01CFH 476 1
01DH							JCC 01E6H 350 1	JCC 01C7H 353 1	JCC 01EAH 160 1	JCC 01E9H 195 1	JCR 01DEH 462 1	JCR 01DDH 472 1	JCR 01D8H 159 1	JCR 01DFH 473 1	JCC 015EH 463 1	JCC 019FH 474 1
01EH							JCR 01E7H 351 1	JCC 01D7H 352 1	JZR 000FH 161 1	JCC 01F9H 197 1			JCR 01E3H 452 1	JCC 017DH 453 1		
01FH						JCR 01FCH 450 1			JCC 008AH 199 1	JCR 01FBH 198 1			JCC 01ECH 451 1	JCC 008DH 135 1		

```
XMAS VERS 2.0 CORRELATION MICROCODE ..............................                ERRORS= 0    PAGE 31

CROSS REFERENCE DIRECTORY

LABEL     REFERENCES

ANX       316,(317)
APX       316,(318)
BLACK     212,(222)
BLACK2    461,(472)
COMMON    (201)
CONT1     423,(424)
CONT2     425,(427)
CONTX2    322,(323)
CONTXX    320,(321)
CORR      57,(80),80
CROSS     56,(438)
CT1       145,(146)
CT2       147,(150)
CX1       368,(369)
CX2       370,(371)
FETCH     58,59,60,61,71,81,161,221,233,239,328,371,384,429,471,477,(478),478
FRM       56,(242)
IMG       57,(94)
LOOP1     (422),423
LOOP2     425,(426)
LP1       (143),145
LP1X      (144),148
LP2       147,(148,
LX1       (366),368
LX2       (367),370
LXXP      (305),320
LXXP2     (306),322
LXXX      (423),426
MIN       56,(70),70
MNV       57,(400)
NOT1      56,(58)
NOT12     57,(71)
NOT16     57,(81)
NOT2      56,(59)
NOT3      56,(60)
NOT6      56,(61)
NRDY      383,(385)
RDY       383,(384)
REF       57,(340)
STQ       56,(275)
WAT       57,(380),386
WHITE     212,(213)
WHITE2    461,(462)
WIN       56,(170)
XX2       385,(386)
```

(LISTING #3)

```
$$$$$$$$  $$$$$$$$  $$$$$ $  $$$$$$$$  $$$$$$$$    $$      $$
   $$        $$     $$    $        $$        $$   $$$$    $$$$
   $$        $$     $$   $$$       $$        $$  $$  $$  $$  $$
   $$        $$     $$  $ $$       $$       $$   $$   $$ $$   $$
   $$        $$     $$ $$ $$       $$      $$    $$   $$ $$   $$
   $$        $$     $$ $  $$   $$$$$$   $$$$     $$   $$ $$   $$
   $$        $$     $$$   $$       $$  $$        $$   $$ $$   $$
   $$       $$$     $     $$       $$  $$    $$  $$   $$ $$   $$
   $$        $$     $ $$$$$  $$$$$$$$   $$$$$$   $$   $$ $$   $$
```

```
XMAS VERS 2.0                                                      ERRORS= 0    PAGE 1

        $LISTFILE=2
        $BITS
        $CROSSREF
        $IMAGE
        $WIDTH=132
        $TITLE='CORRELATION MICROCODE .............................'
        $DISPLAY(ALL)
BITS=1              LEFT=1              FORMS=0             IMAGE=1             LINES=60            PRINT=1
RIGHT=72
TITLE='CORRELATION MICROCODE .............................'
WIDTH=132           CROSSREF=1          LISTFILE=2          SOURCEFILE=2        MICROMEMORY=512


XMAS VERS 2.0 CORRELATION MICROCODE .............................                            ERRORS= 0    PAGE 2

RECORD              CPE    FT  FO    JUMP      KBUSS    PAGEF  CBUS  PAUSEF  STRB  LPCNT    EMIT     MULTIPL  SPRF
NUMBER            6543210  10  10  6543210  76543210    10    210     0       0   43210  76543210     0       0

     1
     2  KBUSS      FIELD  LENGTH=8             DEFAULT=0
     3                           MICROPS(K00000=00H K00001=01H K00002=02H
     4                           K00003=03H K00004=04H
     5                           K00005=05H K00006=06H
     6                           K00007=07H K00008=08H
     7                           K00009=09H K0000A=0AH
     8                           K0000B=0BH K0000C=0CH
     9                           K0000D=0DH  K0000E=0EH
    10                           K0000F=0FH K00010=10H
```

```
                        K00011=11H K00012=12H
                        K00013=13H K00014=14H
                        K00015=15H K00016=16H
                        K00017=17H K00018=18H
                        K00019=19H K0001A=1AH
                        K0001B=1BH K0001C=1CH
                        K0001D=1DH K0001E=1EH
                        K0001F=1FH K01FFF=3FH
                        K80000=80H K7FFFF=7FH
                        KFFFFF=377Q KFFFFA=0FAH);
KBUSS      KBUS;

PAGEF       FIELD LENGTH=2              DEFAULT=10B
                        MICROPS(PAGE1=00B PAGE2=01B
                        PAGE3=10B PAGE4=11B);

CHUS      . FIELD  LENGTH=3          DEFAULT=0
           MICROPS(NBO=000B    INH=001B    RMW=010B    CNB=011B
                 RIN=100B    ROT=101B    RRM=110B    R.M=111B);

PAUSEF                  FIELD LENGTH=1        DEFAULT=1
                        MICROPS(PAUSE=0);

STRB        FIELD LENGTH=1              DEFAULT=0
                        MICROPS(STROBE=1B);
LPCNT       FIELD LENGTH=5              DEFAULT=00010B
                        MICROPS(C0=00010B LCNT1=00100B LCNT2=01000B
                        LCNT3=01100B LCNT4=10000B ITCNT1=00111B
                        ITCNT2=01011B ITCNT3=01111B ITCNT4=10011B);

EMIT        FIELD LENGTH=8              DEFAULT=0
                        MICROPS ( M1=0FFH M2=0FEH M3=0FDH M4=0FCH
                        M5=0FBH M6=0FAH M7=0F9H M8=0F8H M9=0F7H M10=0F6H
                        M11=0F5H M12=0F4H M13=0F3H M14=0F2H M15=0F1H
                        M16=0F0H M17=0EFH M18=0EEH);

MULTIPLY FIELD LENGTH=1                 DEFAULT=0
                        MICROPS(MULT=1B);

SPRF        FIELD  LENGTH=1             DEFAULT=0
                        MICROPS(SPARE=01H);
```

```
XMAS VERS 2.0 CORRELATION MICROCODE .............................                    ERRORS= 0   PAGE 3

RECORD            CPE     FT  FO   JUMP      KRUSS     PAGEF  CBUS  PAUSEF  STRB  LPCNT    EMIT     MULTIPL  SPRF
NUMBER          6543210   10  10  6543210  76543210     10    210     0      0    43210  76543210     0       0

   55
   56
   57  /* COR COR COR COR COR COR  */
   58
   59
   60  /* TEMPLATE MATCHING ALGORITHIM */
   61  /*R0=-153 R7=AT
   62           R8=AR
   63    R2=-145 R5=RP
   64     T=SUM  R4=AC
   65    R1=WORKING
   66    AC=WORKING
   67    R9=WORKING */
   68
   69  /* LOAD ADDRESS POINTERS */
   70  07FH!      COR!   CLP(T);
       (007FH)  1001100  11  00  0001000  00000000    10    000     1      0    00010  00000000     0       0

   71  08FH!             LMI(T) K00010 RRM FF1;
       (008FH)  0011100  11  11  0001001  00010000    10    110     1      0    00010  00000000     0       0

   72  09FH!             ACM(AC);
       (009FH)  0001011  11  00  0001010  00000000    10    000     1      0    00010  00000000     0       0

   73  0AFH!             SDR(R2) FF1; /* R2=TP */
       (00AFH)  0100010  11  11  0001011  11111111    10    000     1      0    00010  00000000     0       0
```

```
74  0BEH!                LMI(T) RRM FF1; /* T=T+1 */
    (00BEH)  0011100  11  11  0001100  00000000   10   110   1   0   00010  00000000   0   0

75  0CEH!                ACM(AC);
    (00CEH)  0001011  11  00  0001101  00000000   10   000   1   0   00010  00000000   0   0

76  0DEH!                SDR(R5) FF1;  /* R5=RP */
    (00DEH)  0100101  11  11  0001110  11111111   10   000   1   0   00010  00000000   0   0

77  0EEH!                LMI(T) FF1 RRM;
    (00EEH)  0011100  11  11  0111100  00000000   10   110   1   0   00010  00000000   0   0

78  0ECH!                ACM(AC);
    (00ECH)  0001011  11  00  0001111  00000000   10   000   1   0   00010  00000000   0   0

79. 0FCH!                SDR(R6) FF1;  /*R6=CP */
    (00FCH)  0100110  11  11  0001101  11111111   10   000   1   0   00010  00000000   0   0
```

```
XMAS VERS 2.0 CORRELATION MICROCODE ..............................              ERRORS= 0    PAGE 4

RECORD              CPE     FT  FO   JUMP      KRUSS    PAGEF  CBUS  PAUSFF  STRB  LPCNT   EMIT     MULTIPL  SPRF
NUMBER            6543210   10  10  6543210  76543210    10    210     0      0    43210  76543210     0       0

  80   /* INITALIZ  LOOP COUNTERS AND LOAD REGISTERS */
  81   0DCH!                ILR(R2) LCNT1 M10: /* AC=IP */
       (00DCH)   0000010   11  00  0010101  00000000    10    000     1      0    00100  11110110     0       0

  82   15CH!                SDR(R7) FF1 LCNT2 M10: /* R7=AI=IP */
       (015CH)   0100111   11  11  0010110  11111111    10    000     1      0    01000  11110110     0       0

  83   /* INITALIZ  CONSTANTS */
  84   16CH!                CLR(T) LCNT3 M10:
       (016CH)   1001100   11  00  0010111  00000000    10    000     1      0    01100  11110110     0       0

  85   17CH!                LMI(T) K0001F FF1 LCNT4 M10:
       (017CH)   0011100   11  11  0011000  00011111    10    000     1      0    10000  11110110     0       0

  86   18CH!                LMI(T) FF1 RRM:
       (018CH)   0011100   11  11  0011001  00000000    10    110     1      0    00010  00000000     0       0

  87   19CH!                ACM(AC):
       (019CH)   0001011   11  00  0011100  00000000    10    000     1      0    00010  00000000     0       0

  88   1CCH!                SDR(R0) FF1:  /* R0=-153 */
       (01CCH)   0100000   11  11  0011101  11111111    10    000     1      0    00010  00000000     0       0

  89   1DCH!                LMI(T) RRM:
       (01DCH)   0011100   11  00  0011111  00000000    10    110     1      0    00010  00000000     0       0
```

```
90  1FCH!          ACM(AC);
    (01FCH)  0001011  11  00  0011110  00000000  10   000   1   0   00010  00000000   0   0

91  1FCH!          SDR(R2) FF1 JZR(LOOP); /* R2=-145 */
    (01ECH)  0100010  11  11  0101110  11111111  10   000   1   0   00010  00000000   0   0

92  /* ENTRY POINT FOR NEW CORRELATION VALUE */
93  00EH!     LOOP! CLR(T) ; /* SUM=0 */
    (000EH)  1001100  11  00  0111101  00000000  10   000   1   0   00010  00000000   0   0

94  00DH!     LOOPP! ILR(R5); /* RP TO AC */
    (000DH)  0000101  11  00  0000011  00000000  10   000   1   0   00010  00000000   0   0

95  03DH!          SDR(R8) FF1 ; /* R8=RP */
    (003DH)  0101000  11  11  0111010  11111111  10   000   1   0   00010  00000000   0   0
```

```
RECORD              CPE     FI FO   JUMP     KBUSS    PAGEF CBUS PAUSFF STRB LPCNT   EMIT    MULTIPL SPRF
NUMBER            6543210   10 10  6543210  76543210   10   210    0      0  43210 76543210    0      0

  96  03AH!       LPINR! LMI(R7) FF1 RRM;
      (003AH)   0010111   11 11  0000100  00000000   10   110    1      0  00010 00000000    0      0

  97  04AH!       LPINR2! ACM(AC)    ; /* C(AI) TO AC */
      (004AH)   0001011   11 00  0111001  00000000   10   000    1      0  00010 00000000    0      0

  98  049H!               LMI(R8) FF1 RRM ITCNT1;
      (0049H)   0011000   11 11  0000011  00000000   10   110    1      0  00111 00000000    0      0

  99  039H!               AIA(T) MULT K01FFF JFL(LPINR,CONON); /* T=T+(I-5)**2*/
      (0039H)   0111110   11 00  1000011  00111111   10   000    1      0  00010 00000000    1      0

 100  03BH! CONON! LMI(R7) K00008 ITCNT2;
      (003BH)   0010111   11 00  0111110  00001000   10   000    1      0  01011 00000000    0      0

 101  03FH!               LMI(R7) FF1 RRM LCNT1 M10 JFL(LPINR2,GOON);
      (003FH)   0010111   11 11  1000100  00000000   10   110    1      0  00100 11110110    0      0

 102  04BH!       GOON!   ILR(T) ;
      (004BH)   0001100   11 00  0111100  00000000   10   000    1      0  00010 00000000    0      0

 103  04CH!               LMI(R6) FF1 RWM LCNT2 M10 JFL(LOPOP,GOON2);
      (004CH)   0010110   11 11  1000101  00000000   10   111    1      0  01000 11110110    0      0
```

```
104  05AH!      LOPOP! ILR(R0);
     (005AH)  0000000  11  00  0111001  00000000  10  000  1  0  00010  00000000  0  0

105  059H!             ALR(R7) JZR(LOOP);
     (0059H)  0000111  11  00  0101110  11111111  10  000  1  0  00010  00000000  0  0

106  05BH!      GOON2! ILR(R2) ITCNT4;
     (005BH)  0000010  11  00  0111111  00000000  10  000  1  0  10011  00000000  0  0

107  05FH!             ALR(R7) LCNT3 M10 JFL(GOLP,END);
     (005FH)  0000111  11  00  1000111  11111111  10  000  1  0  01100  11110110  0  0

108  07AH!      GOLP!  CLR(R1) JZR(LOOPP);
     (007AH)  1000001  11  00  0101101  00000000  10  000  1  0  00010  00000000  0  0

109  07BH!      END!   NOP(AC) PAGE1 JZR(FETCH);
     (007BH)  1101101  11  00  0101111  00000000  00  000  1  0  00010  00000000  0  0
```

```
RECORD            CPE    FI FO   JUMP     KRUSS    PAGEF CBUS PAUSEF STRB LPCNT   EMIT     MULTIPL SPRF
NUMBER          6543210  10 10  6543210  76543210   10   210    0     0   43210  76543210     0     0

 110
 111
 112
 113
 114
 115  /*MICROCODE TO LOCATE CORRELATION MIN*/
 116
 117
 118  /* R1=T */
 119  /* R2=MAX */
 120  /* R0=IM-3 */
 121  /* R5=JM-3 */
 122  /* R6=COR+K */
 123  /* R7=I-3 */
 124  /* R8=J-3 */
 125  /* R9=MIN */
 126  076H!      MIN!   CLR(R6) PAGE3 LCNT1 M10!
      (0076H)  1000110  11 00  0010111  00000000   10   000    1     0   00100  11110110     0     0

 127  176H!           LMI(R6) K00012 RRM LCNT2 M10!
      (0176H)  0010110  11 00  0001000  00010010   10   110    1     0   01000  11110110     0     0

 128  086H!            ACM(AC)!
      (0086H)  0001011  11 00  0001001  00000000   10   000    1     0   00010  00000000     0     0

 129  096H!            SDR(R6) FF1!
      (0096H)  0100110  11 11  0001100  11111111   10   000    1     0   00010  00000000     0     0
```

```
130  /* INITALIZE REGISTERS */
131  0C6H!        CLR(AC) ;
     (00C6H)  1001101  11  00  0001101  00000000   10   000   1   0   00010  00000000   0   0

132  0D6H!              LMI(AC) K00004;
     (00D6H)  0011101  11  00  0001110  00000100   10   000   1   0   00010  00000000   0   0

133  0E6H!              CMR(AC);/* -5 TO AC */
     (00E6H)  1111101  11  00  0001111  00000000   10   000   1   0   00010  00000000   0   0

134  0F6H!              SDR(R0) FF;;/*IM-5 TO R0 */
     (00F6H)  0100000  11  11  0011101  11111111   10   000   1   0   00010  00000000   0   0

135  1D6H!              SDR(R5) FF;;/*JM-5 TO R5 */
     (01D6H)  0100101  11  11  0011100  11111111   10   000   1   0   00010  00000000   0   0

136  1C6H!              SDR(R7) FF;;/* I-5 TO R7 */
```

```
XMAS VERS 2.0 CORRELATION MICROCODE ..............................                                 ERRORS= 0   PAGE 7

RECORD                  CPF    FI  FO   JUMP     KRUSS    PAGEF  CBUS  PAUSEF  STRB  LPCNT    EMIT    MULTIPL  SPRF
NUMBER              6543210   10  10  6543210  76543210   10    210     0      0    43210  76543210    0       0

        (0106H)     0100111   11  11  0010110  11111111   10    000     1      0    00010  00000000    0       0

 137  166H!                   SDR(R8) FF1:/*J-5 TO R8 */
      (0166H)       0101000   11  11  0010000  11111111   10    000     1      0    00010  00000000    0       0

 138  106H!                  LMI(R6) RRM:
      (0106H)       0010110   11  00  0010001  00000000   10    110     1      0    00010  00000000    0       0

 139  116H!                  ACM(AC):
      (0116H)       0001011   11  00  0010010  00000000   10    000     1      0    00010  00000000    0       0

 140  126H!                  SDR(R9) FF1:/* C(COR) TO R9*/
      (0126H)       0101001   11  11  0010011  11111111   10    000     1      0    00010  00000000    0       0

 141  136H!                  SDR(R2) FF1: /*C(COR) TO R2*/
      (0136H)       0100010   11  11  0010100  11111111   10    000     1      0    00010  00000000    0       0

 142  146H!                  CLR(R1):
      (0146H)       1000001   11  00  0001010  00000000   10    000     1      0    00010  00000000    0       0

 143  046H!                 LMI(R1) K00008:
      (0046H)       0010001   11  00  0000001  00001000   10    000     1      0    00010  00000000    0       0
```

```
144  /* BEGIN COMPUTATION LOOP*/
145  016H!      LOOP2! LMI(R8) FF1;  /* J=J+1 */
     (0016H)  0011000  11  11  0000010  00000000   10   000   1   0   00010  00000000   0   0

146  026H!      LOOP1! LMI(R7) FF1;  /* I=I+1 */
     (0026H)  0010111  11  11  0000011  00000000   10   000   1   0   00010  00000000   0   0

147  036H!             LMI(R6) FF1 RRM;
     (0036H)  0010110  11  11  0000100  00000000   10   110   1   0   00010  00000000   0   0

148  /* TEST FOR MAXIMUM */
149  046H!             LCM(T);  /*COMP OF DATA TO T */
     (0046H)  1111010  11  00  0000101  00000000   10   000   1   0   00010  00000000   0   0

150  056H!             ILR(R2);
     (0056H)  0000010  11  00  0000110  00000000   10   000   1   0   00010  00000000   0   0

151  066H!             ADR(T) FF1;
```

```
XMAS VERS 2.0 CORRELATION MICROCODE ..............................                  ERRORS= 0   PAGE 8

RECORD                CPE     FI  FO    JUMP      KBUSS    PAGEF  CBUS  PAUSFF  STRB  LPCNT    EMIT    MULTIPL  SPRF
NUMBER              6543210   10  10  6543210   76543210    10    210     0      0    43210  76543210     0      0

          (0066H)   0111100   11  11  0110101   11111111    10    000     1      0    00010  00000000     0      0

 152  065H!                   NOP(AC) RRM JFL(MLTD,MGED);
          (0065H)   1101101   11  00  1000001   00000000    10    110     1      0    00010  00000000     0      0

 153  012H!            MLTD!  ACM(AC);
          (0012H)   0001011   11  00  0110100   00000000    10    000     1      0    00010  00000000     0      0

 154  014H!                  SDR(R2) FF1 ITCNT1 JZR(NODE);
          (0014H)   0100010   11  11  0101001   11111111    10    000     1      0    00111  00000000     0      0

 155  013H!            MGED!  LCM(T);
          (0013H)   1111010   11  00  0110101   00000000    10    000     1      0    00010  00000000     0      0

 156  015H!                   ILR(R9);   /* AC=MIN */
          (0015H)   0001001   11  00  0000010   00000000    10    000     1      0    00010  00000000     0      0

 157  025H!                  ADR(T) FF1;
          (0025H)   0111100   11  11  0000011   11111111    10    000     1      0    00010  00000000     0      0

 159  035H!                   NOP(AC) RRM JFL(DGTMN,DLEMN);
          (0035H)   1101101   11  00  1000010   00000000    10    110     1      0    00010  00000000     0      0
```

```
159  023H!       QLEMN! TZR(T) FF0;
     (0023H)  1011100  11  00  0110100  11111111   10   000   1   0   00010  00000000   0   0

160  024H!              NOP(AC) RRM JFL(ZZRO,NTZZRO) ;
     (0024H)  1101101  11  00  1000011  00000000   10   110   1   0   00010  00000000   0   0

161  022H!       DGTMN! NOP(R1) ITCNTI JZR(NODE);
     (0022H)  1100001  11  00  0101001  00000000   10   000   1   0   00111  00000000   0   0

162  032H!       Z RO!  NOP(T);
     (0032H)  1101100  11  00  0110100  00000000   10   000   1   0   00010  00000000   0   0

163  034H!              NOP(T);
     (0034H)  1101100  11  00  0000110  00000000   10   000   1   0   00010  00000000   0   0

164  064H!       TLR(R7); /* I-5 TO AC */
     (0064H)  0000111  11  00  0001000  00000000   10   000   1   0   00010  00000000   0   0
```

```
XMAS VERS 2.0 CORRELATION MICROCODE ............................          ERRORS= 0   PAGE 9

RECORD               CPE     FI  FO   JUMP     KBUSS    PAGEF  CBUS  PAUSFF  STRB  LPCNT   EMIT     MULTIPL  SPRF
NUMBER             6543210   10  10  6543210  76543210   10    210     0      0    43210  76543210     0       0

165   /* TAKE ABSOLUTE VALUE */
166   084H!      TZR(AC) K80000 INH;
      (0084H)  1011101   11  00  0001001  10000000   10    001     1      0    00010  00000000     0       0

167   094H!            NOP(T) JFL(POSS,NEGG);
      (0094H)  1101100   11  00  1001010  00000000   10    000     1      0    00010  00000000     0       0

168   0A2H!      POSS!  NOP(AC) JCR(OVER);
      (00A2H)  1101101   11  00  0110101  00000000   10    000     1      0    00010  00000000     0       0

169   0A3H!      NEGG!  CMR(AC);
      (00A3H)  1111101   11  00  0110100  00000000   10    000     1      0    00010  00000000     0       0

170   0A4H!            INR(AC) FF1;
      (00A4H)  0111101   11  11  0110101  00000000   10    000     1      0    00010  00000000     0       0

171   0A5H!      -VER!  SDR(T) FF1; /*ABS(I-5) TO T */
      (00A5H)  0101100   11  11  0001011  11111111   10    000     1      0    00010  00000000     0       0

172   085H!             ILR(R8);/* J-5 TO AC */
      (0085H)  0001000   11  00  0001101  00000000   10    000     1      0    00010  00000000     0       0

173   /* TAKE ABSOLUTE VALUE */
174   0D5H!            TZR(AC) K80000 INH;
      (00D5H)  1011101   11  00  0001110  10000000   10    001     1      0    00010  00000000     0       0
```

```
175  0E5H!            NOP(AC) JFL(POS2,NEG2);
     (00E5H)  1101101  11  00  1001011  00000000   10    000    1    0    00010  00000000    0    0

176  082H!     POS2!  ALR(T) JCR(OV2);  /* ABS(J-3)+ABS(I-3) TO AC,T*/
     (0082H)  0001100  11  00  0110100  11111111   10    000    1    0    00010  00000000    0    0

177  083H!     NEG2!  CMR(AC);
     (0083H)  1111101  11  00  0110001  00000000   10    000    1    0    00010  00000000    0    0

178  081H!            ALR(T) FF1;  /*ABSJ-5)+ABS(I-5)  TO AC,T */
     (0081H)  0001100  11  11  0110100  11111111   10    000    1    0    00010  00000000    0    0

179  084H!     0,2!   ILR(R1);
     (0084H)  0000001  11  00  0001100  00000000   10    000    1    0    00010  00000000    0    0
```

```
XMAS VERS 2.0 CORRELATION MICROCODE .............................          ERRORS= 0   PAGE 10

RECORD                 CPE     FI  FO    JUMP      KRUSS    PAGEF  CBUS  PAUSFF  STRB  LPCNT    EMIT     MULTIPL  SPRF
NUMBER               6543210   10  10  6543210  76543210    10     210     0      0    43210  76543210     0       0

  180  0C4H!                   CMP(AC);
          (00C4H)    1111101   11  00  0001101  00000000    10     000     1      0    00010  00000000     0       0

  181  0D4H!                   ALR(T) FF1 INH;
          (00D4H)    0001100   11  11  0001110  11111111    10     001     1      0    00010  00000000     0       0

  182  0E4H!                   NOP(AC) JFL(NTLTO .NTGEO );
          (00E4H)    1101101   11  00  1001000  00000000    10     000     1      0    00010  00000000     0       0

  183  082H!             NTLTO! ILR(T);
          (0082H)    0001100   11  00  0110000  00000000    10     000     1      0    00010  00000000     0       0

  184  080H!                   SDR(R1) FF1 ; /* T=N TO R1*/
          (0080H)    0100001   11  11  0110001  11111111    10     000     1      0    00010  00000000     0       0

  185  081H!                   ILR(R7);
          (0081H)    0000111   11  00  0000001  00000000    10     000     1      0    00010  00000000     0       0

  186  011H!                   SDR(R0) FF1;  /* IM-5 TO I-5*/
          (0011H)    0100000   11  11  0000010  11111111    10     000     1      0    00010  00000000     0       0

  187  021H!                   ILR(R8);
          (0021H)    0001000   11  00  0000101  00000000    10     000     1      0    00010  00000000     0       0
```

```
188  051H!               SDR(R5) FF1 ITCNT1 JZR(NODE); /* JM=J */
     (0051H)  0100101  11  11  0101001  11111111   10   000   1   0   00111  00000000   0   0

189  083H!        NTGEO! NOP(AC) ITCNT1 JZR(NODE);
     (0083H)  1101101  11  00  0101001  00000000   10   000   1   0   00111  00000000   0   0

190  033H!        NTZZRO! ACM(AC);
     (0033H)  0001011  11  00  0110111  00000000   10   000   1   0   00010  00000000   0   0

191  037H!               SDR(R9) FF1;
     (0037H)  0101001  11  11  0000001  11111111   10   000   1   0   00010  00000000   0   0

192  017H!               ILR(R7);
     (0017H)  0000111  11  00  0000010  00000000   10   000   1   0   00010  00000000   0   0

193  027H!               SDR(R0) FF1; /*IM-5=I-5 */
     (0027H)  0100000  11  11  0000100  11111111   10   000   1   0   00010  00000000   0   0
```

```
XMAS VERS 2.0 CORRELATION MICROCODE ..............................             ERRORS= 0   PAGE 11

RECORD               CPE     FI  FO   JUMP      KBUSS    PAGEF  CBUS  PAUSFF  STRB  LPCNT    EMIT     MULTIPL  SPRF
NUMBER             6543210   10  10  6543210  76543210    10    210     0      0    43210  76543210     0      0

 194   047H!                ILR(R8);
       (0047H)     0001000   11  00  0110011  00000000    10    000     1      0    00010  00000000     0      0

 195   043H!           SDR(R5) FF1;/* JM-5=J-5 */
       (0043H)     0100101   11  11  0001100  11111111    10    000     1      0    00010  00000000     0      0

 196   0C3H!                ILR(R0);  /*IM-3 TO AC */
       (00C3H)     0000000   11  00  0111111  00000000    10    000     1      0    00010  00000000     0      0

 197   0CFH!                TZR(AC) K80000 IVH;
       (00CFH)     1011101   11  00  0010000  10000000    10    001     1      0    00010  00000000     0      0

 198   10FH!                NOP(AC) JFL(POSSS,NEGGG);
       (010FH)     1101101   11  00  1000000  00000000    10    000     1      0    00010  00000000     0      0

 199   109H!           NEGGG! CMR(AC);
       (0109H)     1111101   11  00  0110101  00000000    10    000     1      0    00010  00000000     0      0

 200   105H!                INR(AC) FF1;
       (0105H)     0111101   11  11  0111010  00000000    10    000     1      0    00010  00000000     0      0

 201   10AH!           POSSS! SDR(T) FF1; /* ABS(IM-5) TO T */
       (010AH)     0111100   11  11  0111110  11111111    10    000     1      0    00010  00000000     0      0
```

```
202  10EH!        NOP(T);
     (010EH)  1101100  11  00  0001111  00000000   10   000   1   0   00010  00000000   0   0

203  0FEH!               ILR(R5); /* JM-5 TO AC */
     (00FEH)  0000101  11  00  0111011  00000000   10   000   1   0   00010  00000000   0   0

204  0FBH!               TZR(AC) K80000 IVH;
     (00FBH)  1011101  11  00  0110101  10000000   10   001   1   0   00010  00000000   0   0

205  0F5H!        NOP(AC) JFL(POS1,NEG1);
     (00F5H)  1101101  11  00  1001111  00000000   10   000   1   0   00010  00000000   0   0

206  0F3H!        NEG1! CMR(AC);
     (00F3H)  1111101  11  00  0110100  00000000   10   000   1   0   00010  00000000   0   0

207  0F4H!               INR(AC) FF1;
     (00F4H)  0111101  11  11  0110010  00000000   10   000   1   0   00010  00000000   0   0
```

```
XMAS VERS 2.0 CORRELATION MICROCODE ..............................                ERRORS= 0   PAGE 12

RECORD              CPE     FI  FO   JUMP     KRUSS    PAGEF  CBUS  PAUSFF  STRB  LPCNT    EMIT     MULTIPL  SPRF
NUMBER            6543210   10  10  6543210  76543210   10    210     0      0    43210  76543210     0       0

 208  0F2H!         POSI! ALR(T)! /* TM=ABS(IM-5)+ABS(JM-5) TO AC.T */
      (00F2H)     0001100   11  00  0110001  11111111   10    000     1      0    00010  00000000     0       0

 209  /* TEST FOR I=9 */
 210  0F1H!               SDR(R1) FF1 ITCNT1 JZR(NODE)!
      (00F1H)     0100001   11  11  0101001  11111111   10    000     1      0    00111  00000000     0       0

 211  009H!         NODE! NOP(AC) JFL(INE2.IEQ2)!
      (0009H)     1101101   11  00  1000110  00000000   10    000     1      0    00010  00000000     0       0

 212  06AH!         INE2! NOP(T)!
      (006AH)     1101100   11  00  0000010  00000000   10    000     1      0    00010  00000000     0       0

 213  02AH!               NOP(T) JCR(LOOP1)!
      (002AH)     1101100   11  00  0110110  00000000   10    000     1      0    00010  00000000     0       0

 214  06BH!         IEQ2! CLR(R7) LCNT1 M10!
      (006BH)     1000111   11  00  0110010  00000000   10    000     1      0    00100  11110110     0       0

 215  062H!               LMI(R7)  K00004!
      (0062H)     0010111   11  00  0110011  00000100   10    000     1      0    00010  00000000     0       0

 216  063H!               CMR(R7) ITCNT2! /* -5 TO R7 */
      (0063H)     1110111   11  00  0111100  00000000   10    000     1      0    01011  00000000     0       0
```

```
217  /* TEST FOR J=9 */
218  06CH!            NOP(AC) JFL(JNE2,JEQ2);
     (006CH)  1101101  11  00  1000001  00000000   10   000   1   0   00010  00000000   0   0

219  01AH!     JNE2!  NOP(T) JCR(LOOP2);
     (001AH)  1101100  11  00  0110110  00000000   10   000   1   0   00010  00000000   0   0

220  01BH!     JEQ2!  NOP(T);
     (001BH)  1101100  11  00  0111100  00000000   10   000   1   0   00010  00000000   0   0

221  /* WRITE RESULTS OT MACRO MEMORY */
222  01CH!            CLR(AC);
     (001CH)  1001101  11  00  0000010  00000000   10   000   1   0   00010  00000000   0   0

223  02CH!            LMI(AC) K00017;
     (002CH)  0011101  11  00  0111011  00010111   10   000   1   0   00010  00000000   0   0
```

```
XMAS VERS 2.0 CORRELATION MICROCODE ..............................            ERRORS= 0   PAGE 13

RECORD              CPE     FI  FO    JUMP      KRUSS     PAGEF  CBUS  PAUSEF  STRB  LPCNT    EMIT     MULTIPL  SPRF
NUMBER            6543210   10  10  6543210   76543210     10    210     0      0    43210  76543210     0       0

 224   02BH!        ILR(R9) RWM;  /* MIN TO AC*/
      (002BH)   0001001   11  00  0111101   00000000     10    111     1      0    00010  00000000     0       0

 225   02DH!                CLR(AC);
      (002DH)   1001101   11  00  0000110   00000000     10    000     1      0    00010  00000000     0       0

 226   06DH!                LMI(AC) K00016;
      (006DH)   0011101   11  00  0001001   00010110     10    000     1      0    00010  00000000     0       0

 227   09DH!                ILR(R2) RWM;  /*MAX TO AC*/
      (009DH)   0000010   11  00  0001110   00000000     10    111     1      0    00010  00000000     0       0

 228   0EDH!                CLR(AC);
      (00EDH)   1001101   11  00  0010011   00000000     10    000     1      0    00010  00000000     0       0

 229   13DH!                LMI(AC) K0001A    ;
      (013DH)   0011101   11  00  0111110   00011010     10    000     1      0    00010  00000000     0       0

 230   13EH!                ILR(R0) RWM;
      (013EH)   0000000   11  00  0111100   00000000     10    111     1      0    00010  00000000     0       0

 231   13CH!                CLR(AC);
      (013CH)   1001101   11  00  0011010   00000000     10    000     1      0    00010  00000000     0       0
```

```
232  1ACH!              LMI(AC) K0001B;
     (01ACH)  0011101  11  00  0011011  00011011  10   000   1   0   00010  00000000   0   0

233  1BCH!             ILR(R5) RWM PAGE1 JZR(FETCH);
     (01BCH)  0000101  11  00  0101111  00000000  00   111   1   0   00010  00000000   0   0

234  00FH!        FETCH! NOP(AC) JZR(FETCH);
     (000FH)  1101101  11  00  0101111  00000000  10   000   1   0   00010  00000000   0   0

235  EOF

NO PROGRAM ERRORS
END OF PROGRAM
```

XMAS VERS 2.0 CORRELATION MICROCODE .... ERRORS= 0 PAGE 14

MICROPROGRAM MEMORY IMAGE

	0H	1H	2H	3H	4H	5H	6H	7H	8H	9H	AH	BH	CH	DH	EH	FH
000H										JFL 006AH 211 5				JCC 003DH 94 2	JCR 000DH 93 2	JZR 000FH 234 3
001H		JCC 0021H 186 1	JCR 0014H 153 1	JCR 0015H 155 1	JZR 0009H 154 1	JCC 0025H 156 1	JCC 0026H 145 2	JCC 0027H 192 1			JCR 0016H 219 1	JCR 001CH 220 1	JCC 002CH 222 1			
002H		JCC 0051H 187 1	JZR 0009H 161 1	JCR 0024H 159 1	JFL 0032H 160 1	JCC 0035H 157 1	JCC 0036H 146 2	JCC 0047H 193 1			JCR 0026H 213 1	JCR 002DH 224 1	JCR 0023H 223 1	JCC 006DH 225 1		
003H			JCR 0034H 162 1	JCR 0037H 190 1	JCC 0064H 163 1	JFL 0022H 158 1	JCC 0046H 147 1	JCC 0017H 191 1		JFL 003AH 99 1	JCC 004AH 96 2	JCR 003FH 100 1		JCR 003AH 95 1	JFL 004AH 101 1	
004H				JCC 00C3H 195 1			JCC 0056H 149 1	JCR 0043H 194 1		JCC 0039H 98 1	JCR 0049H 97 2	JCR 004CH 102 1	JFL 005AH 103 1			
005H		JZR 0009H 188 1					JCC 0066H 150 1			JZR 000FH 105 1	JCR 0059H 104 1	JCR 005FH 106 1				JFL 007AH 107 1
006H			JCR 0063H 215 1	JCR 006CH 216 1	JCC 0084H 164 1	JFL 0012H 152 1	JCC 0055H 151 1				JCC 002AH 212 1	JCR 0062H 214 1	JFL 0014H 218 1	JCC 009DH 226 1		
007H							JCC 0176H 126 0				JZR 000DH 108 1	JZR 000FH 109 1			JCC 008FH 70 0	

XMAS VERS 2.0 CORRELATION MICROCODE .... ERRORS= 0 PAGE 15

MICROPROGRAM MEMORY IMAGE

	0H	1H	2H	3H	4H	5H	6H	7H	8H	9H	AH	BH	CH	DH	EH	FH
008H	JCR 0081H 184 1	JCC 0011H 185 1	JCR 0080H 183 1	JZR 0009H 189 1	JCC 0094H 166 1		JCC 0096H 128 1								JCC 009FH 71 1	
009H					JFL 00A2H 167 1		JCC 00C6H 129 1							JCC 00EDH 227 1	JCC 00AEH 72 1	
00AH			JCR 00A5H 168 1	JCR 00A4H 169 1	JCR 00A5H 170 1	JCC 00B5H 171 2	JCC 0016H 143 1								JCC 00BEH 73 1	
00BH		JCR 0084H 178 1	JCR 00B4H 176 1	JCR 00B1H 177 1	JCC 00C4H 179 2	JCC 00D5H 172 1									JCC 00CEH 74 1	
00CH				JCR 00CFH 196 1	JCC 00D4H 180 1		JCC 00D6H 131 1								JCC 00DEH 75 1	JCC 010FH 197 1
00DH					JCC 00E4H 181 1	JCC 00F5H 174 1	JCC 00E6H 132 1						JCC 015CH 91 1		JCC 00EEH 76 1	
00EH					JFL 00B2H 182 1	JFL 00B2H 175 1	JCC 00F6H 133 1						JCC 00FCH 78 1	JCC 013DH 228 1	JCR 00ECH 77 1	
00FH		JZR 0009H 210 1	JCR 00F1H 208 2	JCR 00F4H 206 1	JCR 00F2H 207 1	JFL 00F2H 205 1	JCC 0106H 134 1					JCR 00F5H 204 1	JCC 00DCH 79 1		JCR 00FBH 203 1	

XMAS VERS 2.0 CORRELATION MICROCODE .... ERRORS= 0 PAGE 16

MICROPROGRAM MEMORY IMAGE

	0H	1H	2H	3H	4H	5H	6H	7H	8H	9H	AH	BH	CH	DH	EH	FH
010H						JCR 010AH 200 1	JCC 0116H 138 1				JCR 010EH 201 2	JCR 0105H 199 1			JCC 00FEH 202 1	JFL 010AH 198 1
011H							JCC 0126H 139 1									
012H							JCC 0136H 140 1									
013H							JCC 0146H 141 1						JCC 014CH 231 1	JCR 013FH 229 1	JCR 013CH 230 1	
014H							JCC 00A6H 142 1									
015H													JCC 016CH 92 1			
016H							JCC 0106H 137 1						JCC 017CH 94 1			
017H							JCC 0086H 127 1						JCC 018CH 95 1			

XMAS VERS 2.0 CORRELATION MICROCODE .... ERRORS= 0 PAGE 17

MICROPROGRAM MEMORY IMAGE

	0H	1H	2H	3H	4H	5H	6H	7H	8H	9H	AH	BH	CH	DH	EH	FH
018H													JCC 019CH 96 1			
019H													JCC 01CCH 97 1			
01AH													JCC 01BCH 232 1			
01BH													JZR 000FH 233 1			
01CH							JCC 0166H 136 1						JCC 01DCH 98 1			
01DH							JCC 01C6H 135 1						JCC 01FCH 99 1			
01EH													JZR 000EH 91 1			
01FH													JCC 01ECH 90 1			

XMAS VERS 2.0 CORRELATION MICROCODE .... ERRORS= 0 PAGE 18

CROSS REFERENCE DIRECTORY

```
LABEL      REFERENCES

CONON      99,(100)
COR        (70)
DGTMN      158,(161)
DLEMN      158,(159)
END        107,(109)
FETCH      109,233,(234),234
GOLP       107,(108)
GOON       101,(102)
GOON2      103,(106)
IEQ2       211,(214)
INE2       211,(212)
JEQ2       218,(220)
JNE2       218,(219)
LOOP       91,(93),105
LOOP1      (146),213
LOOP2      (145),219
LOOPP      (94),108
LOPOP      103,(104)
LPINR      (96),99
LPINR2     (97),101
MGED       152,(155)
MIN        (126)
MLTD       152,(153)
NEG1       205,(206)
NEG2       175,(177)
NEGG       167,(169)
NEGGG      198,(199)
NODE       154,161,188,189,210,(211)
NTGED      182,(189
NTLTD      182,(183)
NTZZRO     160,(190)
OV2        176,(179
OVER       168,(171)
POS1       205,(208)
POS2       175,(176)
POSS       167,(168)
POSSS      198,(201)
ZZRO       160,(162)
```

APPENDIX C

Pacer Emulation Microcode (Page 1)

(LISTING #4)

```
XMAS VERS 2.0                                                        ERRORS= 0   PAGE 1

      $LISTFILE=2
      $CROSSREF
      $BITS
      $IMAGE
      $WIDTH=132
      $TITLE='PACER EMULATOR MICROCODE FOR INTEL 3000'
      $DISPLAY(ALL)
BITS=1            LEFT=1            FORMS=0            IMAGE=1            LINES=60            PRINT=1
RIGHT=72
TITLE='PACER EMULATOR MICROCODE FOR INTEL 3000'
WIDTH=132         CROSSREF=1        LISTFILE=2         SOURCEFILE=2       MICROMEMORY=512


XMAS VERS 2.0 PACER EMULATOR MICROCODE FOR INTEL 3000                                  ERRORS= 0   PAGE 2

RECORD            CPE      FI  FO    JUMP      KBUSS    PAGEF  CBUS  PAUSFF  STRB  LPCNT    EMIT     MULTF  SPRF
NUMBER         6543210     10  10  6543210   76543210    10    210     0      0   43210  76543210    0     0

   1
   2  KBUSS       FIELD  LENGTH=8          DEFAULT=0
   3                        MICROPS(K00000=00H K00001=01H K00002=02H
   4                        K00003=03H K00004=04H
   5                        K00005=05H K00006=06H
   6                        K00007=07H K00008=08H
   7                        K00009=09H K0000A=0AH
   8                        K0000B=0BH K0000C=0CH
   9                        K0000D=0DH  K0000E=0EH
  10                        K0000F=0FH K00010=10H
  11                        K00011=11H K00012=12H
  12                        K00013=13H K00014=14H
  13                        K00015=15H K00016=16H
  14                        K00017=17H K00018=18H
```

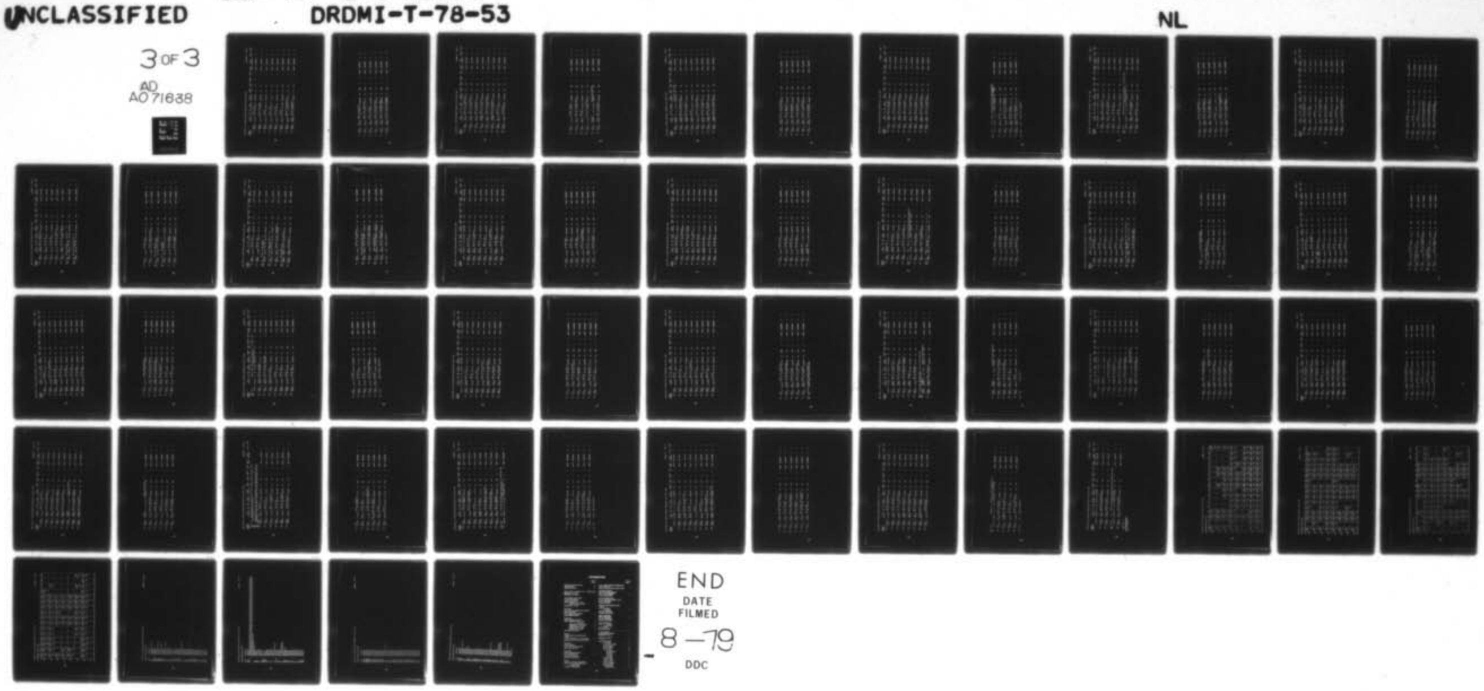
```
                                K0001F=1FH K01FFF=3FH
                                K00019=19H K0001A=1AH
                                K0001B=1BH K0001C=1CH
                                K0001D=1DH K0001E=1EH
                                K80000=80H K7FFFF=7FH
                                KFFFFF=377Q KFFFFA=0FAH);
KBUSS           KBUS;

/* ALL MICROCODE IS ON PAGE 1 */
PAGEF           FIELD LENGTH=2                  DEFAULT=0
                                MICROPS(PAGE1=00B PAGE2=01B
                                PAGE3=10B PAGE4=11B);

CBUS            FIELD  LENGTH=3             DEFAULT=0
                MICROPS(NBO=000B   INH=001B   RMW=010B   CNB=011B
                        RIN=100B   ROT=101B   RRM=110B   RWM=111B);

/*              NBO     NOBUS OPERATION
                INH     INHIBIT CPE ARRAY
                RMW     READ-MODIFY-WRITE
                CNB     CPU NEEDS BUS
                RIN     REQUEST INPUT
                ROT     REQUEST OUTPUT
                RRM     REQUEST READ MEMORY
                RWM     REQUEST WRITE MEMORY
                                                                    */

PAUSEF          FIELD LENGTH=1                  DEFAULT=1B
                                MICROPS(SP0=0);

STRB            FIELD LENGTH=1                  DEFAULT=0
                                MICROPS(STROBE=1B);

LPCNT           FIELD LENGTH=5                  DEFAULT=00010B
                                MICROPS(C0=00010B LCNT1=00100B LCNT2=01000B
                                LCNT3=01100B LCNT4=10000B ITCNT1=00111B
                                ITCNT2=01011B ITCNT3=01111B ITCNT4=10011B);

EMIT            FIELD LENGTH=8                  DEFAULT=0
                                MICROPS  (M1=0FFH M2=0FEH M3=0FDH M4=0FCH
```

```
XMAS VERS 2.0 PACER EMULATOR MICROCODE FOR INTEL 3000                                         ERRORS= 0   PAGE 3

RECORD           CPE     FI  FO    JUMP      KBUSS    PAGEF  CBUS  PAUSFF  STRB  LPCNT    EMIT     MULTF  SPRF
NUMBER         6543210   10  10  6543210   76543210    10    210     0      0    43210  76543210     0      0

  55                              M5=0FBH M6=0FAH M7=0F9H M8=0F8H M9=0F7H M10=0F6H
  56                              M11=0F5H M12=0F4H M13=0F3H M14=0F2H M15=0F1H
  57                              M16=0F0H M17=0FFH M18=0FEH);
  58
  59  MULTF FIELD LENGTH=1                        DEFAULT=08
  60                         MICROPS(MULT=18);
  61
  62  SPRF        FIELD  LENGTH=1                 DEFAULT=0
  63                              MICROPS(SPARE=01H);
  64
  65  A           STRING 'R0';
  66  X           STRING 'R1';
  67  Q           STRING 'R2';
  68  P           STRING 'R3';
  69  S           STRING 'R4';
  70  B           STRING 'R5';
  71  E           STRING 'R6';
  72  W           STRING 'R7';
  73
  74  00H!        INIT!  CLR(A);
      (0000H)  1000000  11  00  0000001   00000000    00    000     1      0    00010  00000000     0      0

  75  10H!               CLR(X);
      (0010H)  1000001  11  00  0000010   00000000    00    000     1      0    00010  00000000     0      0

  76  20H!               CLR(W) JCC(INIT1);
      (0020H)  1000111  11  00  0000100   00000000    00    000     1      0    00010  00000000     0      0
```

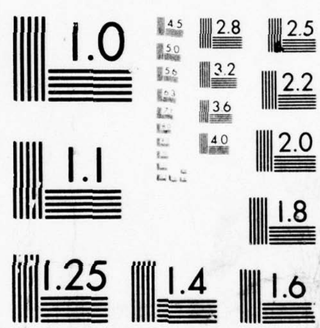
```
77  30H!      DIR!   LMI(AC) RRM JPR(LA STA LX STX AI SI 4 D AOM ORI XORI
78                                 AVDD C SZ SNZ GG)!
    (0030H)  0011101  11  00  1100111  00000000   00    110    1    0    00010  00000000    0    0

79  40H!      INIT1! CLR(T) JCC(INIT2)!
    (0040H)  1001100  11  00  0001001  00000000   00    400    1    0    00010  00000000    0    0

80  60H!      SE!    NOP(A) JCF(NTEQL.EQL)!
    (0060H)  1100000  11  00  1010010  00000000   00    400    1    0    00010  00000000    0    0

81  70H!      LA!    ACM(AC)!
    (0070H)  0001011  11  00  0001000  00000000   00    400    1    0    00010  00000000    0    0

82  80H!             SDR(A) FFI JZR(FETCH)!
    (0080H)  0100000  11  11  0101111  11111111   00    400    1    0    00010  00000000    0    0

83  90H!      INIT2! LMI(T)!
    (0090H)  0011100  11  00  0001010  00000000   00    400    1    0    00010  00000000    0    0
```

AD-A071 638 ARMY MISSILE RESEARCH AND DEVELOPMENT COMMAND REDSTO--ETC F/G 9/2
APPLICATION OF A MICROPROGRAMMED, BIT SLICE MICROPROCESSOR TO T--ETC(U)
SEP 78 L G MINOR
UNCLASSIFIED DRDMI-T-78-53 NL

3 OF 3
AD
A071638



END
DATE
FILMED
8-79
DDC



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

```
XMAS VERS 2.0 PACER EMULATOR MICROCODE FOR INTEL 3000                                      ERRORS= 0   PAGE 4

RECORD                 CPE     FI  FO   JUMP     KBUSS    PAGEF  CBUS  PAUSEF  STRB  LPCNT   EMIT     MULTF  SPRF
NUMBER               6543210   10  10  6543210  76543210   10    210     0      0    43210  76543210    0     0

   84  0A0H!                  ILR(W) ROT;
          (00A0H)    0000111   11  00  0001011  00000000   00    101     1      0    00010  00000000    0     0

   85  0B0H!                  LMI(T) FF1 RRM;
          (00B0H)    0011100   11  11  0001100  00000000   00    110     1      0    00010  00000000    0     0

   86  0C0H!                  ACM(AC);
          (00C0H)    0001011   11  00  0001101  00000000   00    000     1      0    00010  00000000    0     0

   87  0D0H!                  SDR(S) FF1;
          (00D0H)    0100100   11  11  0001110  11111111   00    000     1      0    00010  00000000    0     0

   88  0E0H!                  LMI(T) RRM;
          (00E0H)    0011100   11  00  0001111  00000000   00    110     1      0    00010  00000000    0     0

   89  0F0H!                  ACM(AC);
          (00F0H)    0001011   11  00  0010000  00000000   00    000     1      0    00010  00000000    0     0

   90  100H!                  SDR(P) FF1 JZR(FETCH);
          (0100H)    0100011   11  11  0101111  11111111   00    000     1      0    00010  00000000    0     0

   91  110H!           CLR1!  CLR(A) JZR(FETCH);
          (0110H)    1000000   11  00  0101111  00000000   00    000     1      0    00010  00000000    0     0
```

```
92  1E0H!      NI!      NOP(A) JZR(FETCH)!
    (01E0H)  1100000  11  00  0101111  00000000   00    000    1    0    00010  00000000   0    0

93  31H!       J!       SDR(P) FF1 JZR(FETCH)!
    (0031H)  0100011  11  11  0101111  11111111   00    000    1    0    00010  00000000   0    0

94  61H!       SG!      NOP(A) JCC(SGC)!
    (0061H)  1100000  11  00  0001001  00000000   00    000    1    0    00010  00000000   0    0

95  71H!       STA!     ILR(A) RWM JZR(FETCH)!
    (0071H)  0000000  11  00  0101111  00000000   00    111    1    0    00010  00000000   0    0

96  91H!       SGC!     NOP(A) JCF(NTFQL2.EQL2)!
    (0091H)  1100000  11  00  1010001  00000000   00    000    1    0    00010  00000000   0    0

97  111H!      A.A!     ILR(A) FF1 STC JZR(FETCH)!
    (0111H)  0000000  10  11  0101111  00000000   00    000    1    0    00010  00000000   0    0
```

```
XMAS VERS 2.0 PACER EMULATOR MICROCODE FOR INTEL 3000                                        ERRORS= 0   PAGE 5

RECORD              CPE     FI  FO   JUMP     KBUSS     PAGEF  CBUS  PAUSFF  STRB  LPCNT     EMIT    MULTF  SPRF
NUMBER            6543210   10  10  6543210  76543210    10    210     0      0    43210  76543210    0      0

  98  131H!        YY!    SDR(A) FF1 JZR(FETCH);
      (0131H)   0100000   11  11  0101111  11111111    00    000     1      0    00010  00000000    0      0

  99  151H!        WW!    SDR(A) FF1 JZR(FETCH);
      (0151H)   0100000   11  11  0101111  11111111    00    000     1      0    00010  00000000    0      0

 100  1F1H!        N2!    NOP(A) JZR(FETCH);
      (01F1H)   1100000   11  00  0101111  00000000    00    000     1      0    00010  00000000    0      0

 101  02H!         NTEQ3! NOP(A) JZF(LT,NTLT);
      (0002H)   1100000   11  00  1011001  00000000    00    000     1      0    00010  00000000    0      0

 102  12H!         LT!    ILR(P) FF1 JZR(FETCH);
      (0012H)   0000011   11  11  0101111  00000000    00    000     1      0    00010  00000000    0      0

 103  22H!         NTEQL! NOP(A) JZR(FETCH);
      (0022H)   1100000   11  00  0101111  00000000    00    000     1      0    00010  00000000    0      0

 104  32H!         L!     LMI(S) FF1;
      (0032H)   0010100   11  11  0000100  00000000    00    000     1      0    00010  00000000    0      0

 105  42H!                ILR(P) RWM;
      (0042H)   0000011   11  00  0000101  00000000    00    111     1      0    00010  00000000    0      0
```

```
106  52H!               ILR(R9) JCR(CL);
     (0052H)  0001001  11  00  0110101  00000000  00  000  1  0  00010  00000000  0  0

107  62H!      SL!      NOP(A) JCF(NTEQ3+EQ3);
     (0062H)  1100000  11  00  1010000  00000000  00  000  1  0  00010  00000000  0  0

108. 72H!      LX!      ACM(AC);
     (0072H)  0001011  11  00  0001000  00000000  00  000  1  0  00010  00000000  0  0

109  82H!               SDR(X) FF1 JZR(FETCH);
     (0082H)  0100001  11  11  0101111  11111111  00  000  1  0  00010  00000000  0  0

110  92H!      NTEQL2!                     NOP(A) JZF(NTGT+GT);
     (0092H)  1100000  11  00  1011010  00000000  00  000  1  0  00010  00000000  0  0

111  0A2H!     NTGT!    NOP(A) JZR(FETCH);
     (00A2H)  1100000  11  00  0101111  00000000  00  000  1  0  00010  00000000  0  0
```

```
XMAS VERS 2.0 PACER EMULATOR MICROCODE FOR INTEL 3000                                    ERRORS= 0   PAGE 6

RECORD              CPF     FI  FO   JUMP      KRUSS    PAGEF  CBUS  PAUSFF  STRB  LPCNT    EMIT     MULTF  SPRF
NUMBER             6543210  10  10  6543210  76543210    10    210     0      0    43210  76543210     0     0

 112  0D2H!       N_EQL4!                   ILR(P) FF1 JZR(FETCH);
      (00D2H)    0000011  11  11  0101111  00000000    00    000     1      0    00010  00000000     0     0

 113  0E2H!       NTEQ5! NOP(A) JZF(NTGT5,GT5);
      (00E2H)    1100000  11  00  1011111  00000000    00    000     1      0    00010  00000000     0     0

 114  0F2H!       NTGT5! NOP(A) JZR(FETCH);
      (00F2H)    1100000  11  00  0101111  00000000    00    000     1      0    00010  00000000     0     0

 115  112H!       CA0!   CLR(A);
      (0112H)    1000000  11  00  0010010  00000000    00    000     1      0    00010  00000000     0     0

 116  122H!              ILR(A) FF1 JZR(FETCH);
      (0122H)    0000000  11  11  0101111  00000000    00    000     1      0    00010  00000000     0     0

 117  132H!       OUT2!  ALR(AC) FFZ JCR(YY);
      (0132H)    0001101  11  10  0110001  11111111    00    000     1      0    00010  00000000     0     0

 118  142H!       OUT3!  NOP(A) JZR(FETCH);
      (0142H)    1100000  11  00  0101111  00000000    00    000     1      0    00010  00000000     0     0

 119  162H!       NTEQ6! NOP(A) JZF(NTGT6,GT6);
      (0162H)    1100000  11  00  1011111  00000000    00    000     1      0    00010  00000000     0     0
```

```
I20  172H!     NTGT6! ILR(P) FF1 JZR(FETCH);
     (0172H)   0000011  11  11  0101111  00000000   00   000   1   0   00010  00000000   0   0

I21  1F2H!        N3!    NOP(A) JZR(FETCH);
     (01F2H)   1100000  11  00  0101111  00000000   00   000   1   0   00010  00000000   0   0

I22  03H!       E03!     NOP(A) JZR(FETCH);
     (0003H)   1100000  11  00  0101111  00000000   00   000   1   0   00010  00000000   0   0

I23  13H!      NTLT!     NOP(A) JZR(FETCH);
     (0013H)   1100000  11  00  0101111  00000000   00   000   1   0   00010  00000000   0   0

I24  23H!       EOL!     ILR(P) FF1 JZR(FETCH);
     (0023H)   0000011  11  11  0101111  00000000   00   000   1   0   00010  00000000   0   0

I25  33H!       REG!     ILR(A) JCC(REGC);
     (0033H)   0000000  11  00  0010000  00000000   00   000   1   0   00010  00000000   0   0
```

```
RECORD                CPE      FI  FO   JUMP     KBUSS    PAGEF  CBUS  PAUSFF  STRB  LPCNT    EMIT     MULTF  S2RF
NUMBER              6543210    10  10  6543210  76543210   10    210     0      0    43210  76543210    0      0

 126  63H!            SNE!    NOP(A) JCC(SVEC);
     (0063H)        1100000    11  00  0001000  00000000   00    000     1      0    00010  00000000    0      0

 127  73H!            STX!    ILR(X) RWM JZR(FETCH);
     (0073H)        0000001    11  00  0101111  00000000   00    111     1      0    00010  00000000    0      0

 128  83H!            SNEC!   NOP(A) JCF(NTEQL4+EQL4);
     (0083H)        1100000    11  00  1010101  00000000   00    000     1      0    00010  00000000    0      0

 129  93H!            EQL2!   NOP(A) JZR(FETCH);
     (0093H)        1100000    11  00  0101111  00000000   00    000     1      0    00010  00000000    0      0

 130  0A3H!           GT!     ILR(P) FF1 JZR(FETCH);
     (00A3H)        0000011    11  11  0101111  00000000   00    000     1      0    00010  00000000    0      0

 131  0D3H!           EQL4!   NOP(A) JZR(FETCH);
     (00D3H)        1100000    11  00  0101111  00000000   00    000     1      0    00010  00000000    0      0

 132  0E3H!           EQ5!    ILR(P) FF1 JZR(FETCH);
     (00E3H)        0000011    11  11  0101111  00000000   00    000     1      0    00010  00000000    0      0

 133  0F3H!           GT5!    ILR(P) FF1 JZR(FETCH);
     (00F3H)        0000011    11  11  0101111  00000000   00    000     1      0    00010  00000000    0      0
```

```
134  123H!      R-GC!  NOP(A) JPR(CLR1 ADA CAO TCA ARS ALS LRS SSP
135                            SSN EX EP ES ICX DCX NOMB PZ)!
     (0103H)  1100000  11  00  1100001  00000000  00   000   1   0   00010  00000000   0   0

136  113H!      TCA!   CMP(A)!
     (0113H)  1110000  11  00  0010010  00000000  00   000   1   0   00010  00000000   0   0

137  123H!             ILR(A) FF1 STC JZR(FETCH)!
     (0123H)  0000000  10  11  0101111  00000000  00   000   1   0   00010  00000000   0   0

138  133H!      AGN2!  DSM(R9) JCR(XX)!
     (0133H)  0011001  11  00  0110100  11111111  00   000   1   0   00010  00000000   0   0

139  143H!      AGN3!  DSM(R9) JCR(BACK)!
     (0143H)  0011001  11  00  0110101  11111111  00   000   1   0   00010  00000000   0   0

140  163H!      E06!   ILR(P) FF1 JZR(FETCH)!
```

```
XMAS VERS 2.0 PACER EMULATOR MICROCODE FOR INTEL 3000                                          ERRORS= 0   PAGE 8

RECORD                 CPE      FI  FO    JUMP     KBUSS     PAGEF  CBUS  PAUSFF  STRB  LPCNT    EMIT     MULTF  SPRF
NUMBER               6543210    10  10  6543210  76543210     10    210     0      0    43210  76543210    0      0

         (0163H)     0000011    11  11  0101111  00000000     00    000     1      0    00010  00000000    0      0

141   173H!       GT6!     NOP(A) JZR(FETCH);
         (0173H)     1100000    11  00  0101111  00000000     00    000     1      0    00010  00000000    0      0

142   1E3H!       N4!      NOP(A) JZR(FETCH);
         (01E3H)     1100000    11  00  0101111  00000000     00    000     1      0    00010  00000000    0      0

143   34H!        I!       LMI(AC) RRM;
         (0034H)     0011101    11  00  0000100  00000000     00    110     1      0    00010  00000000    0      0

144   44H!                 ACM(AC);
         (0044H)     0001011    11  00  0000101  00000000     00    000     1      0    00010  00000000    0      0

145   54H!                 LMI(AC) RRM JPR(LA STA LX STX A1 S1 M D AOM OR1 XOR1
146                                  ANDD C SZ SNZ GG);
         (0054H)     0011101    11  00  1100111  00000000     00    110     1      0    00010  00000000    0      0

147   64H!        SGE!     NOP(A) JCC(SGEC);
         (0064H)     1100000    11  00  0001110  00000000     00    000     1      0    00010  00000000    0      0

148   74H!        A1!      ACM(AC);
         (0074H)     0001011    11  00  0001000  00000000     00    000     1      0    00010  00000000    0      0
```

```
[49 84H!                ALR(A) STC JZR(FETCH)!
    (0084H)  0000000  10  00  0101111  11111111  00   000   1   0   00010  00000000   0   0

[50 0E4H!      SGEC!   NOP(A) JCF(NTEU5.EQ5)!
    (00E4H)  1100000  11  00  1010110  00000000  00   000   1   0   00010  00000000   0   0

[51 114H!      APS!    TZP(A) STC K80000 INH!
    (0114H)  1010000  10  00  0010010  10000000  00   001   1   0   00010  00000000   0   0

[52 124H!              DSM(R9)!
    (0124H)  0011001  11  00  0010011  11111111  00   000   1   0   00010  00000000   0   0

[53 134H!      XX!     SRA(AC) STZ FFC JFL(OUT2.AGN2)!
    (0134H)  0001111  01  01  1000011  00000000  00   000   1   0   00010  00000000   0   0

[54 1D4H!      DWN1!   LMI(R9) FF1 JCR(PUSA4)!
    (01D4H)  0011001  11  11  0111001  00000000  00   000   1   0   00010  00000000   0   0
```

```
XMAS VERS 2.0 PACER EMULATOR MICROCODE FOR INTEL 3000                                          ERRORS= 0   PAGE 9

RECORD              CPE     FI  FO    JUMP      KBUSS    PAGEF  CBUS  PAUSFF  STRB  LPCNT     EMIT     MULTF  SPRF
NUMBER            6543210   10  10  6543210   76543210    10    210     0      0    43210   76543210    0      0

 155  1E4H!        N5!     NOP(A) JZR(FETCH);
      (01F4H)    1100000   11  00  0101111   00000000    00    000     1      0    00010   00000000    0      0

 156  35H!         JI!     LMI(AC) RRM;
      (0035H)    0011101   11  00  0000100   00000000    00    110     1      0    00010   00000000    0      0

 157  45H!                 ACM(AC);
      (0045H)    0001011   11  00  0000101   00000000    00    000     1      0    00010   00000000    0      0

 158  55H!         CL!     SDR(P) FF1 JZR(FETCH);
      (0055H)    0100011   11  11  0101111   11111111    00    000     1      0    00010   00000000    0      0

 159  65H!         SLE!    NOP(A) JCC(SLEC);
      (0065H)    1100000   11  00  0010101   00000000    00    000     1      0    00010   00000000    0      0

 160  75H!         SI!     ACM(AC);
      (0075H)    0001011   11  00  0001000   00000000    00    000     1      0    00010   00000000    0      0

 161  85H!                 CIA(AC) FF1;
      (0085H)    0011111   11  11  0001001   00000000    00    000     1      0    00010   00000000    0      0

 162  95H!                 ALR(A) STC JZR(FETCH);
      (0095H)    0000000   10  00  0101111   11111111    00    000     1      0    00010   00000000    0      0
```

```
163  115H!        ALS!     DSM(R9);
     (0115H)  0011001  11  00  0010010  11111111  00  000  1  0  00010  00000000  0  0

164  125H!                 ILR(A);
     (0125H)  0000000  11  00  0010011  00000000  00  000  1  0  00010  00000000  0  0

165  135H!                 DSM(R9);
     (0135H)  0011001  11  00  0010100  11111111  00  000  1  0  00010  00000000  0  0

166  145H!        BACK!    ALR(A) STC JFL(OUT3,AGN3);
     (0145H)  0000000  10  00  1000100  11111111  00  000  1  0  00010  00000000  0  0

167  155H!        SLEC!    NOP(A) JCF(NTEQ6,EQ6);
     (0155H)  1100000  11  00  1010110  00000000  00  000  1  0  00010  00000000  0  0

168  105H!        DWN2!    ILR(R8) RWM JCR(DWN1);
     (0105H)  0001000  11  00  0110100  00000000  00  111  1  0  00010  00000000  0  0
```

```
RECORD                  CPE     FI  FO    JUMP      KBUSS    PAGEF  CBUS  PAUSFF  STRB  LPCNT     EMIT    MULTF  SPRF
NUMBER                6543210  10  10  6543210  76543210    10    210     0      0    43210  76543210    0      0

 169  1E5H!       TRAP!  NOP(A) JZR(FETCH);
      (01E5H)  1100000  11  00  0101111  00000000    00    000     1      0    00010  00000000    0      0

 170  36H!        LT!    LMI(S) FF1;
      (0036H)  0010100  11  11  0000100  00000000    00    000     1      0    00010  00000000    0      0

 171  46H!               ILR(P) RWM;
      (0046H)  0000011  11  00  0000101  00000000    00    111     1      0    00010  00000000    0      0

 172  56H!               LMI(R9) RRM JCR(CLI);
      (0056H)  0011001  11  00  0110111  00000000    00    110     1      0    00010  00000000    0      0

 173  66H!        PUSA!  LMI(R9) FF1 JCC(PUSAC);
      (0066H)  0011001  11  11  0001011  00000000    00    000     1      0    00010  00000000    0      0

 174  /*   LOAD MULTIPLIER    ---MLT--%T      */
 175  76H!        M!     ACM(T);
      (0076H)  0001010  11  00  0001000  00000000    00    000     1      0    00010  00000000    0      0

 176  /*   SAVE SIGN BIT IN C FLAG     */
 177  86H!               ILR(T);
      (0086H)  0001100  11  00  0111110  00000000    00    000     1      0    00010  00000000    0      0
```

```
178  8EH!              SDR(R9) FF1;
     (008EH)  0101001  11  11  0001001  11111111  00  000  1  0  00010  00000000  0  0

179  9EH!              TZR(A) K80000 INH STC;
     (009EH)  1010000  10  00  0110110  10000000  00  001  1  0  00010  00000000  0  0

180  /*     WIPE OUT SIGN BIT     */
181  96H!              TZR(T) K7FFFF;
     (0096H)  1011100  11  00  0001010  01111111  00  000  1  0  00010  00000000  0  0

182  /*   SET " LOOP COUNTER      */
183  0A6H!             CLR(AC);
     (00A6H)  1001101  11  00  0110101  00000000  00  000  1  0  00010  00000000  0  0

184  0A5H!             LMI(AC) K0000F;
     (00A5H)  0011101  11  00  0110100  00001111  00  000  1  0  00010  00000000  0  0

185  0A4H!             LMI(AC) K00004;
     (00A4H)  0011101  11  00  0001011  00000100  00  000  1  0  00010  00000000  0  0
```

```
XMAS VERS 2.0 PACER EMULATOR MICROCODE FOR INTEL 3000                                       ERRORS= 0   PAGE 11

RECORD                CPE    FI  FO    JUMP     KBUSS    PAGEF  CBUS  PAUSEF  STRB  LPCNT    EMIT    MULTF  SPRF
NUMBER              6543210  10  10  6543210  76543210    10    210     0      0    43210  76543210    0      0

 186  0B4H!                  CIA(AC) FF1;
      (00B4H)       0011111  11  11  0110101  00000000    00    000     1      0    00010  00000000    0      0

 187  0B5H!                  SDR(R8) FF1;
      (00B5H)       0101000  11  11  0001100  11111111    00    000     1      0    00010  00000000    0      0

 188  /*    CLEAR PARTIAL PRODUCT    */
 189  0C5H!                  CLR(AC);
      (00C5H)       1001101  11  00  0110100  00000000    00    000     1      0    00010  00000000    0      0

 190  /*    FETCH AND TEST MULTIPLIER    LSB    */
 191  0C4H!                  SRA(T);
      (00C4H)       0001110  11  00  0110010  00000000    00    000     1      0    00010  00000000    0      0

 192  0C2H!          MLP!    LMI(R8) FF1 STZ JFL(MBZ,MB1);
      (00C2H)       0011000  01  11  1001011  00000000    00    000     1      0    00010  00000000    0      0

 193  /*   ADD SEQUENCE    */
 194  0B3H!          MB1!    SDR(R7) FF1 JCR(OV1);
      (00B3H)       0100111  11  11  0111111  11111111    00    000     1      0    00010  00000000    0      0

 195  0BFH!          OV1!    ILR(A) FF0;
      (00BFH)       0000000  11  00  0111110  00000000    00    000     1      0    00010  00000000    0      0
```

```
196  08EH!              ALR(R7) FF0 JCR(MRZ);
     (008EH)  0000111  11  00  0110010  11111111   00   000   1   0   00010  00000000   0   0

197  /*    SHIFT RIGHT FILL WITH ONES    */
198  0B2H!      MRZ!    SRA(AC) FFC STZ;
     (00B2H)  0001111  01  01  0110001  00000000   00   000   1   0   00010  00000000   0   0

199  0B1H!              SRA(T) FFZ JZF(MLP,MEX);
     (00B1H)  0001110  11  10  1011100  00000000   00   000   1   0   00010  00000000   0   0

200  /*        APPLY CORRECTION  */
201  0C3H!      MEX!    TZR(R9) K80000 INH JCR(XXX);
     (00C3H)  1011001  11  00  0111001  10000000   00   001   1   0   00010  00000000   0   0

202  0C9H!      XXX!    NOP(A) JFL(POS,NEG);
     (00C9H)  1100000  11  00  1001100  00000000   00   000   1   0   00010  00000000   0   0

203  0CBH!      NEG!    CMR(A) JCR(OV2);
```

```
XMAS VERS 2.0 PACER EMULATOR MICROCODE FOR INTEL 3000                                    ERRORS= 0   PAGE 12

RECORD               CPE     FI  FO    JUMP     KBUSS    PAGEF  CBUS  PAUSFF  STRB  LPCNT    EMIT     MULTF  SPRF
NUMBER             6543210   10  10  6543210  76543210    10    210     0      0    43210  76543210    0      0

         (00CBH)   1110000   11  00  0111101  00000000    00    000     1      0    00010  00000000    0      0

  204    0CDH!      OV2!    ALR(A) FF1;
         (00CDH)   0000000   11  11  0111010  11111111    00    000     1      0    00010  00000000    0      0

  205    0CAH!      POS!   SDR(A) FF1 JCR(OV3);
         (00CAH)   0100000   11  11  0111110  11111111    00    000     1      0    00010  00000000    0      0

  206    0CEH!      OV3!    ILR(T);
         (00CEH)   0001100   11  00  0110001  00000000    00    000     1      0    00010  00000000    0      0

  207    0C1H!              SRA(AC) FF0;
         (00C1H)   0001111   11  00  0111111  00000000    00    000     1      0    00010  00000000    0      0

  208    0CFH!              SDR(O) FF1 JZR(FETCH);
         (00CFH)   0100010   11  11  0101111  11111111    00    000     1      0    00010  00000000    0      0

  209    0B6H!    PUSAC!    ILR(R0) RWM;
         (00B6H)   0000000   11  00  0001100  00000000    00    111     1      0    00010  00000000    0      0

  210    0C6H!               LMI(R9) FF1;
         (00C6H)   0011001   11  11  0001101  00000000    00    000     1      0    00010  00000000    0      0
```

```
211  0D6H!             ILR(R1) RWM;
     (00D6H)  0000001  11  00  0001110  00000000  00  111  1  0  00010  00000000  0  0

212  0E6H!             LMI(R9) FFI;
     (00E6H)  0011001  11  11  0001111  00000000  00  000  1  0  00010  00000000  0  0

213  0F6H!             ILR(R2) RWM;
     (00F6H)  0000010  11  00  0010000  00000000  00  111  1  0  00010  00000000  0  0

214  106H!             LMI(R9) FFI JCC(PUS42);
     (0106H)  0011001  11  11  0010011  00000000  00  000  1  0  00010  00000000  0  0

215  116H!    LPS!     ILR(A);
     (0116H)  0000000  11  00  0010010  00000000  00  000  1  0  00010  00000000  0  0

216  126H!             DSM(R9) JCC(ZZ);
     (0126H)  0011001  11  00  0010101  11111111  00  000  1  0  00010  00000000  0  0
```

XMAS VERS 2.0 PACER EMULATOR MICROCODE FOR INTEL 3000 ERRORS= 0 PAGE 13

RECORD NUMBER		CPE	FI	FO	JUMP	KBUSS	PAGEF	CBUS	PAUSFF	STRB	LPCNT	EMIT	MULTF	SPRF
		6543210	10	10	6543210	76543210	10	210	0	0	43210	76543210	0	0
217	136H!	PUSA2!	ILR(R3)		RWM!									
	(0136H)	0000011	11	00	0010100	00000000	00	111	1	0	00010	00000000	0	0
218	146H!		LMI(R9)		FF1 JCC(PUSA3)!									
	(0146H)	0011001	11	11	0010110	00000000	00	000	1	0	00010	00000000	0	0
219	156H!	ZZ!	SRA(AC)		STZ JFL(OUT,AGN)!									
	(0156H)	0001111	01	00	1000101	00000000	00	000	1	0	00010	00000000	0	0
220	152H!	OUT!	ALR(AC)		FFZ JCR(WW)!									
	(0152H)	0001101	11	10	0110001	11111111	00	000	1	0	00010	00000000	0	0
221	153H!	AGN!	DSM(R9)		JCR(ZZ)!									
	(0153H)	0011001	11	00	0110110	11111111	00	000	1	0	00010	00000000	0	0
222	166H!	PUSA3!	ILR(R4)		RWM!									
	(0166H)	0000100	11	00	0010111	00000000	00	111	1	0	00010	00000000	0	0
223	176H!		LMI(R9)		FF1!									
	(0176H)	0011001	11	11	0011000	00000000	00	000	1	0	00010	00000000	0	0
224	186H!		ILR(R5)		RWM!									
	(0186H)	0000101	11	00	0011001	00000000	00	111	1	0	00010	00000000	0	0

```
225  196H!              LMI(R9) FF1;
     (01964)  0011001  11  11  0011010  00000000  00  000  1  0  00010  00000000  0  0

226  1A6H!              ILR(R6) RWM;
     (01A6H)  0000110  11  00  0011011  00000000  00  111  1  0  00010  00000000  0  0

227  1R6H!              LMI(R9) FF1;
     (01R6H)  0011001  11  11  0011100  00000000  00  000  1  0  00010  00000000  0  0

228  1C6H!              ILR(R7) RWM;
     (01C6H)  0000111  11  00  0000010  00000000  00  111  1  0  00010  00000000  0  0

229  026H!           LMI(R9) FF1;
     (0026H)  0011001  11  11  0110111  00000000  00  000  1  0  00010  00000000  0  0

230  27H!            ILR(R8) RWM;
     (0027H)  0001000  11  00  0111000  00000000  00  111  1  0  00010  00000000  0  0
```

```
RECORD                CPE     FI  FO   JUMP     KBUSS    PAGEF  CBUS  PAUSEF  STRB  LPCNT    EMIT    MULTF  SPRF
NUMBER              6543210   10  10  6543210  76543210   10    210     0      0    43210  76543210   0      0

 231  28H!       LMI(R9) FF1;
      (0028H)   0011001   11  11  0111001  00000000   00    000     1      0    00010  00000000   0      0

 232  29H!       ILR(T) RWM JZR(FETCH);
      (0029H)   0001100   11  00  0101111  00000000   00    111     1      0    00010  00000000   0      0

 233  1E6H!      N7!    NOP(A) JZR(FETCH);
      (01E6H)   1100000   11  00  0101111  00000000   00    000     1      0    00010  00000000   0      0

 234  1D6H!             LMI(R9) FF1 JCR(DWN2);
      (01D6H)   0011001   11  11  0110101  00000000   00    000     1      0    00010  00000000   0      0

 235  37H!       MSC!   NOP(A) JPR(SF SG SL SNE SGE SLE PUSA PUSX POPX
 236                              RTN SKV SKP SO SNO SAE LAIBUS);
      (0037H)   1100000   11  00  1100110  00000000   00    000     1      0    00010  00000000   0      0

 237  57H!       CLI!   LTM(AC) JCR(CL);
      (0057H)   1011011   11  00  0110101  11111111   00    000     1      0    00010  00000000   0      0

 238  67H!       PUSX!  LMI(S) FF1 JCC(PUSX1);
      (0067H)   0010100   11  11  0001000  00000000   00    000     1      0    00010  00000000   0      0

 239  /*     DIVIDE ..........................................          */
 240  77H!       DI     CLR(R6);
      (0077H)   1000110   11  00  0001111  00000000   00    000     1      0    00010  00000000   0      0
```

```
241  0F7H!              CLR(AC)$
     (00F7H)  1001101  11  00  0010000  00000000  00  000  1  0  00010  00000000  0  0

242  107H!              LMI(AC) K0000F$
     (0107H)  0011101  11  00  0010010  00001111  00  000  1  0  00010  00000000  0  0

243  127H!              LMI(AC) K00005$
     (0127H)  0011101  11  00  0010011  00000101  00  000  1  0  00010  00000000  0  0

244  137H!              CIA(AC) FF1$
     (0137H)  0011111  11  11  0010100  00000000  00  000  1  0  00010  00000000  0  0

245  147H!              SDR(R8) FF1$
     (0147H)  0101000  11  11  0010101  11111111  00  000  1  0  00010  00000000  0  0
```

```
XMAS VERS 2.0 PACER EMULATOR MICROCODE FOR INTEL 3000                                       ERRORS= 0    PAGE 15

RECORD                CPE     FI  FO   JUMP      KRUSS    PAGEF  CBUS  PAUSFF  STRB  LPCNT    EMIT    MULTF  SPRF
NUMBER              6543210   10  10  6543210  76543210    10    210     0       0    43210  76543210    0      0

 246   157H!                 LMI(R9) RRM;
       (0157H)     0011001   11  00  0010110  00000000    00    110     1       0    00010  00000000    0      0

 247   167H!                 ACM(AC);
       (0167H)     0001011   11  00  0010111  00000000    00    000     1       0    00010  00000000    0      0

 248   177H!                 SDR(R9) FF1;
       (0177H)     0101001   11  11  0011000  11111111    00    000     1       0    00010  00000000    0      0

 249   187H!                 SDR(T) FF1;
       (0187H)     0101100   11  11  0110101  11111111    00    000     1       0    00010  00000000    0      0

 250   185H!                 ILR(A);
       (0185H)     0000000   11  00  0110100  00000000    00    000     1       0    00010  00000000    0      0

 251   184H!                 SDR(R2) FF1;
       (0184H)     0100010   11  11  0110010  11111111    00    000     1       0    00010  00000000    0      0

 252   /*  START MAIN LOOP, COMPARE SIGN Y WITH SIGN RI     */
 253   182H!         DIV!    XNR(R9);
       (0182H)     1111001   11  00  0110001  11111111    00    000     1       0    00010  00000000    0      0

 254   181H!                 TZR(R9) FF0 KR0000 INH;
       (0181H)     1011001   11  00  0110000  10000000    00    001     1       0    00010  00000000    0      0
```

```
255  180H!            ILR(R6) JFL(SZERO.SONE);
     (0180H)  0000110  11  00  1001001  00000000   00    000    1    0    00010  00000000   0    0

256  /*   DIFFERENT SIGNS   */
257  192H!      SZERO! ALR(R6) FF0;
     (0192H)  0000110  11  00  0110001  11111111   00    000    1    0    00010  00000000   0    0

258  191H!            ILR(R2);
     (0191H)  0000010  11  00  0110000  00000000   00    000    1    0    00010  00000000   0    0

259  190H!            ALR(R2) FF0;
     (0190H)  0000010  11  00  0011010  11111111   00    000    1    0    00010  00000000   0    0

260  1A0H!            ILR(T);
     (01A0H)  0001100  11  00  0011011  00000000   00    000    1    0    00010  00000000   0    0
```

```
RECORD                CPE_    FI  FO  JUMP     KBUSS     PAGEF  CBUS  PAUSFF  STRB  LPCNT  EMIT      MULTF  SPRF
NUMBER                6543210 10  10  6543210  76543210  10     210   0       0     43210  76543210  0      0

 261  180H!                   SDR(R9) FF1;
      (0180H)         0101001 11  11  0011100  11111111  00     000   1       0     00010  00000000  0      0

 262  1C0H!                   ALR(R2) JCR(OVER);
      (01C0H)         0000010 11  00  0110001  11111111  00     000   1       0     00010  00000000  0      0

 263  /*   SAME SIGNS   */
 264  193H!    SONE!          ALR(R6) FF1;
      (0193H)         0000110 11  11  0110100  11111111  00     000   1       0     00010  00000000  0      0

 265  194H!                   ILR(R2);
      (0194H)         0000010 11  00  0110101  00000000  00     000   1       0     00010  00000000  0      0

 266  195H!                   ALR(R2);
      (0195H)         0000010 11  00  0110111  11111111  00     000   1       0     00010  00000000  0      0

 267  197H!                   ILR(T);
      (0197H)         0001100 11  00  0011010  00000000  00     000   1       0     00010  00000000  0      0

 268  1A7H!                   SDR(R9);
      (01A7H)         0101001 11  00  0011011  11111111  00     000   1       0     00010  00000000  0      0

 269  187H!                   CMR(AC);
      (0187H)         1111101 11  00  0110001  00000000  00     000   1       0     00010  00000000  0      0
```

```
270  191H!          ALR(R2) FF1;
     (0191H)  0000010  11  11  0011100  11111111   00    000    1     0    00010  00000000    0     0

271  /*   CHECK FOR I=0      */
272  1C1H!     OVER!  LMI(R8) FF1;
     (01C1H)  0011000  11  11  0110111  00000000   00    000    1     0    00010  00000000    0     0

273  1C7H!          NOP(A) JFL(DIV.DONE);
     (01C7H)  1100000  11  00  1001000  00000000   00    000    1     0    00010  00000000    0     0

274  /*     APPLY CORRECTION      */
275  183H!     DONE!  CLR(AC);
     (0183H)  1001101  11  00  0111000  00000000   00    000    1     0    00010  00000000    0     0

276  189H!          LMI(AC) K80000 FF1;
     (0189H)  0011101  11  11  0011001  10000000   00    000    1     0    00010  00000000    0     0
```

```
XMAS VERS 2.0 PACER EMULATOR MICROCODE FOR INTEL 3000                              ERRORS= 0   PAGE 17

RECORD                CPE     F1  FO   JUMP      KRUSS     PAGEF  CBUS  PAUSFF  STRB  LPCNT    EMIT     MULTF  SPRF
NUMBER              6543210   10  10  6543210  76543210     10    210     0      0    43210  76543210     0     0

  277  198H!                 ALR(R6);
       (0198H)      0000110   11  00  0011010  11111111     00    000     1      0    00010  00000000     0     0

  278  1A8H!                 SDR(A) FF1 JZR(FETCH);
       (01A8H)      0100000   11  11  0101111  11111111     00    000     1      0    00010  00000000     0     0

  279  87H!          PUSX1!  ILR(A) RWM;
       (0087H)      0000000   11  00  0001001  00000000     00    111     1      0    00010  00000000     0     0

  280  97H!                  LMI(S) FF1;
       (0097H)      0010100   11  11  0001010  00000000     00    000     1      0    00010  00000000     0     0

  281  0A7H!                 ILR(X) RWM;
       (00A7H)      0000001   11  00  0001011  00000000     00    111     1      0    00010  00000000     0     0

  282  0B7H!                 LMI(S) FF1;
       (00B7H)      0010100   11  11  0001100  00000000     00    000     1      0    00010  00000000     0     0

  283  0C7H!                 ILR(P) RWM;
       (00C7H)      0000011   11  00  0001101  00000000     00    111     1      0    00010  00000000     0     0

  284  0D7H!                 LMI(S) FF1;
       (00D7H)      0010100   11  11  0001110  00000000     00    000     1      0    00010  00000000     0     0
```

```
285  0E7H!             ILR(W) RWM JZR(FETCH);
     (00E7H)  0000111  11  00  0101111  00000000   00    111    1    0    00010  00000000   0    0

286  117H!     SSP!    TZR(A) K7FFFF JZR(FETCH);
     (0117H)  1010000  11  00  0101111  01111111   00    000    1    0    00010  00000000   0    0

287  107H!     PUSA6!  ILR(T) RWM JZR(FETCH);
     (0107H)  0001100  11  00  0101111  00000000   00    111    1    0    00010  00000000   0    0

288  1E7H!     RMI!    NOP(A) JZR(FETCH);
     (01E7H)  1100000  11  00  0101111  00000000   00    000    1    0    00010  00000000   0    0

289  38H!      XK!     ILR(X);
     (0038H)  0000001  11  00  0000001  00000000   00    000    1    0    00010  00000000   0    0

290  019H!             ALR(R9);
     (0019H)  0001001  11  00  0000100  11111111   00    000    1    0    00010  00000000   0    0
```

```
XMAS VERS 2.0 PACER EMULATOR MICROCODE FOR INTEL 3000                                    ERRORS= 0   PAGE 18

RECORD               CPE     FI  FO   JUMP      KBUSS    PAGEF  CBUS  PAUSFF  STRB  LPCNT    EMIT     MULTF  S2RF
NUMBER             6543210   10  10  6543210  76543210    10    210     0      0    43210  76543210     0     0

 291   48H!                 LMI(AC) RRM JPR(LA STA LX STX A1 S1 M D AOM OR1 XOR1
 292                                            AVDD C SZ SNZ GG);
       (0048H)     0011101   11  00  1100111  00000000    00    110     1      0    00010  00000000     0     0

 293   68H!          POPX!  DSM(S) JCC(POPX1);
       (0068H)     0010100   11  00  0001000  11111111    00    000     1      0    00010  00000000     0     0

 294   78H!          AOM!   ACM(AC) FF1 RWM JZR(FETCH);
       (0078H)     0001011   11  11  0101111  00000000    00    111     1      0    00010  00000000     0     0

 295   88H!          POPX1! LMI(S) RRM;
       (0088H)     0010100   11  00  0001001  00000000    00    110     1      0    00010  00000000     0     0

 296   98H!                 ACM(AC);
       (0098H)     0001011   11  00  0001010  00000000    00    000     1      0    00010  00000000     0     0

 297   0A8H!                  SDR(W) FF1;
       (00A8H)     0100111   11  11  0001011  11111111    00    000     1      0    00010  00000000     0     0

 298   0B8H!                  DSM(S);
       (00B8H)     0010100   11  00  0001100  11111111    00    000     1      0    00010  00000000     0     0

 299   0C8H!                  LMI(S) RRM;
       (00C8H)     0010100   11  00  0001101  00000000    00    110     1      0    00010  00000000     0     0
```

```
300  0D8H!                ACM(AC)$
     (00D8H)  0001011  11  00  0001110  00000000  00  000  1  0  00010  00000000  0  0

301  0E8H!                SDR(P)FF]$
     (00E8H)  0100011  11  11  0001111  11111111  00  000  1  0  00010  00000000  0  0

302  0F8H!                DSM(S)$
     (00F8H)  0010100  11  00  0010000  11111111  00  000  1  0  00010  00000000  0  0

303  108H!                LMI(S) RRM JCC(POPX2)$
     (0108H)  0010100  11  00  0010011  00000000  00  110  1  0  00010  00000000  0  0

304  118H!       SSN!     CSR(AC)$
     (0118H)  0101101  11  00  0010010  00000000  00  000  1  0  00010  00000000  0  0

305  128H!                ORR(A) K80000 JZR(FETCH)$
```

```
RECORD                 CPE     FI  FO   JUMP     KBUSS    PAGEF  CBUS  PAUSFF  STRB  LPCNT    EMIT    MULTF  SPRF
NUMBER               6543210   10  10  6543210  76543210   10    210     0      0    43210  76543210    0      0

         (012BH)     1100000   11  00  0101111  10000000   00    000     1      0    00010  00000000    0      0

 306  13BH!      POPX2! SDR(X) FF1;
         (013BH)     0100001   11  11  0010100  11111111   00    000     1      0    00010  00000000    0      0

 307  14BH!             DSM(S);
         (014BH)     0010100   11  00  0010101  11111111   00    000     1      0    00010  00000000    0      0

 308  15BH!             LMI(S) RRM;
         (015BH)     0010100   11  00  0010110  00000000   00    110     1      0    00010  00000000    0      0

 309  16BH!             SDR(A) FF1 JZR(FETCH);
         (016BH)     0100000   11  11  0101111  11111111   00    000     1      0    00010  00000000    0      0

 310  1DBH!      PUSA5! LMI(R9) FF1 JCR(PUSA6);
         (01DBH)     0011001   11  11  0110111  00000000   00    000     1      0    00010  00000000    0      0

 311  1EBH!      SMI!   NOP(A) JZR(FETCH);
         (01EBH)     11 0000   11  00  0101111  00000000   00    000     1      0    00010  00000000    0      0

 312  39H!       JJ!    NOP(A) JZR(FETCH);
         (0039H)     1100000   11  00  0101111  00000000   00    000     1      0    00010  00000000    0      0
```

```
313  43H!        RTN2!   ACM(AC)  JCR(RTN3)!
     (0043H)  0001011  11  00  0111010  00000000  00  000  1  0  00010  00000000  0  0

314  59H!        RTN1!   LMI(S)  RRM  JCC(RTN2)!
     (0059H)  0010100  11  00  0000100  00000000  00  110  1  0  00010  00000000  0  0

315  69H!        RTN!    DSM(S)  JCC(RTN1)!
     (0069H)  0010100  11  00  0000101  11111111  00  000  1  0  00010  00000000  0  0

316  79H!        OR1!    ACM(AC)!
     (0079H)  0001011  11  00  0001000  00000000  00  000  1  0  00010  00000000  0  0

317  89H!                ORR(A)  JZR(FETCH)!
     (0089H)  1100000  11  00  0101111  11111111  00  000  1  0  00010  00000000  0  0

318  119H!       EX!     SDR(T)  FF1!
     (0119H)  0101100  11  11  0010010  11111111  00  000  1  0  00010  00000000  0  0
```

```
RECORD                CPE    FI FO   JUMP      KBUSS    PAGEF  CBUS  PAUSFF  STRB  LPCNT    EMIT     MULTF  S2RF
NUMBER              6543210  10 10  6543210  76543210    10    210     0      0    43210  76543210    0      0

  319  129H!                 ILR(X);
       (0129H)     0000001  11 00  0010011  00000000    00    000     1      0    00010  00000000    0      0

  320  139H!                 SDR(A) FF1;
       (0139H)     0100000  11 11  0010100  11111111    00    000     1      0    00010  00000000    0      0

  321  149H!                 ILR(T);
       (0149H)     0001100  11 00  0010101  00000000    00    000     1      0    00010  00000000    0      0

  322  159H!                 SDR(X) FF1 JZR(FETCH);
       (0159H)     0100001  11 11  0101111  11111111    00    000     1      0    00010  00000000    0      0

  323  189H!          DN1!   SDR(T) FF1;
       (0189H)     0101100  11 11  0011001  11111111    00    000     1      0    00010  00000000    0      0

  324  199H!                 ILR(Q);
       (0199H)     0000010  11 00  0011010  00000000    00    000     1      0    00010  00000000    0      0

  325  1A9H!                 SDR(A) FF1;
       (01A9H)     0100000  11 11  0011011  11111111    00    000     1      0    00010  00000000    0      0

  326  1B9H!                 ILR(T);
       (01B9H)     0001100  11 00  0011100  00000000    00    000     1      0    00010  00000000    0      0
```

```
327  1C9H!          SDR(O) FF1 JZR(FETCH);
     (01C9H)  0100010  11  11  0101111  11111111   00   000   1   0   00010  00000000   0   0

328  109H!     PUSA4! ILR(R9) RWM JCR(PUSA5);
     (0109H)  0001001  11  00  0111000  00000000   00   111   1   0   00010  00000000   0   0

329  1E9H!     EQ!    ILR(A) JCC(ON1);
     (01E9H)  0000000  11  00  0011000  00000000   00   000   1   0   00010  00000000   0   0

330  1AH!      POS1!  ILR(P) FF1 JZR(FETCH);
     (001AH)  0000011  11  11  0101111  00000000   00   000   1   0   00010  00000000   0   0

331  2AH!      POSS!  NOP(A) JZR(FETCH);
     (002AH)  1100000  11  00  0101111  00000000   00   000   1   0   00010  00000000   0   0

332  /* ENTRY FOR PAGE 2 CENTROID TRACKER */
333  3AH!      CTRACK! NOP(A) JZR(FETCH) PAGE3;
```

```
RECORD              CPE     FI  FO   JUMP      KBUSS     PAGEF  CBUS  PAUSFF  STRB  LPCNT    EMIT      MULTF  SPRF
NUMBER            6543210  10  10  6543210  76543210     10    210     0      0    43210  76543210     0      0

        (003AH)   1100000  11  00  0101111  00000000     10    000     1      0    00010  00000000     0      0

 334   4AH!       RTN3!  SDR(P) FF1 JZR(FETCH);
        (004AH)   0100011  11  11  0101111  11111111     00    000     1      0    00010  00000000     0      0

 335   5AH!       SKN1!  NOP(A) JFL(POSS.NEGG);
        (005AH)   1100000  11  00  1000010  00000000     00    000     1      0    00010  00000000     0      0

 336   6AH!       SKN!   TZR(A) K80000 INH JCC(SKN1);
        (006AH)   1010000  11  00  0000101  10000000     00    001     1      0    00010  00000000     0      0

 337   7AH!       XOR1!  ACM(AC);
        (007AH)   0001011  11  00  0001000  00000000     00    000     1      0    00010  00000000     0      0

 338   8AH!       XNR(A) JCR(DWN5);
        (008AH)   1110000  11  00  0111101  11111111     00    000     1      0    00010  00000000     0      0

 339   /*  IF A .LT. MEM CF=0 ZF=0, IF A .GT. MEM CF=0 ZF=1
 340     IF A .EQ. MEM CF=1 ZF=0     */
 341   0AAH!      NTSAME!                 NOP(A) JFL(APOS2.ANEG2);
        (00AAH)   1100000  11  00  1001111  00000000     00    000     1      0    00010  00000000     0      0

 342   0BAH!      NOCRY! NOP(A) JZR(FETCH);
        (00BAH)   1100000  11  00  0101111  00000000     00    000     1      0    00010  00000000     0      0
```

```
343  0DAH!      NOCRY2!                    NOP(A) FF1 STC JZR(FETCH);
     (00DAH)  1100000  10  11  0101111  00000000  00   000   1    0   00010  00000000   0    0

344  0FAH!      APOS2! NOP(A) FF1 STZ JZR(FETCH);
     (00FAH)  1100000  01  11  0101111  00000000  00   000   1    0   00010  00000000   0    0

345  10AH!      SP!   ILR(P) FF1 JZR(FETCH);
     (010AH)  0000011  11  11  0101111  00000000  00   000   1    0   00010  00000000   0    0

346  11AH!      EP!   SDR(T)  FF1;
     (011AH)  0101100  11  11  0010010  11111111  00   000   1    0   00010  00000000   0    0

347  12AH!            ILR(P);
     (012AH)  0000011  11  00  0010011  00000000  00   000   1    0   00010  00000000   0    0

348  13AH!            SDR(A) FF1;
```

```
XMAS VERS 2.0 PACER EMULATOR MICROCODE FOR INTEL 3000                                                   ERRORS= 0   PAGE 22

RECORD              CPE     FI  FO    JUMP     KBUSS     PAGEF  CBUS  PAUSFF   STRB  LPCNT     EMIT      MULTF  SPRF
NUMBER            6543210   10  10  6543210  76543210     10    210     0       0    43210   76543210     0      0

         (013AH)  0100000   11  11  0010100  11111111     00    000     1       0    00010   00000000     0      0

 349   14AH!       ILR(T)!
         (014AH)  0001100   11  00  0010101  00000000     00    000     1       0    00010   00000000     0      0

 350   15AH!               SDR(P) FF1 JZR(FETCH)!
         (015AH)  0100011   11  11  0101111  11111111     00    000     1       0    00010   00000000     0      0

 351   16AH!      NSKIP! NOP(A) JZR(FETCH)!
         (016AH)  1100000   11  00  0101111  00000000     00    000     1       0    00010   00000000     0      0

 352   17AH!      EVEN!  ILR(P) FF1 JZR(FETCH)!
         (017AH)  0000011   11  11  0101111  00000000     00    000     1       0    00010   00000000     0      0

 353   18AH!      N-!    NOP(A) JFL(NOVEF1,OVEF1)!
         (018AH)  1100000   11  00  1001001  00000000     00    000     1       0    00010   00000000     0      0

 354   19AH!      NOVEF1!                        NOP(A) JZR(FETCH)!
         (019AH)  1100000   11  00  0101111  00000000     00    000     1       0    00010   00000000     0      0

 355   1AAH!      OVEF2! ILR(P) FF1 JZR(FETCH)!
         (01AAH)  0000011   11  11  0101111  00000000     00    000     1       0    00010   00000000     0      0
```

```
356  1BAH!      NC2!   NOP(A) JFL(NOVEF3,OVEF3);
     (01BAH)  1100000  11  00  1001100  00000000  00  000  1  0  00010  00000000  0  0

357  1CAH!      NOVEF3!                 ILR(P) FF1  JZR(FETCH);
     (01CAH)  0000011  11  11  0101111  00000000  00  000  1  0  00010  00000000  0  0

358  1DAH!      0_EF4! ILR(P) FF1 JZR(FETCH);
     (01DAH)  0000011  11  11  0101111  00000000  00  000  1  0  00010  00000000  0  0

359  1EAH!      N11!   NOP(A) JZR(FETCH);
     (01EAH)  1100000  11  00  0101111  00000000  00  000  1  0  00010  00000000  0  0

360  1BH!       NEG1!  NOP(A) JZR(FETCH);
     (001BH)  1100000  11  00  0101111  00000000  00  000  1  0  00010  00000000  0  0

361  2BH!       NEGG!  ILR(P) FF1 JZR(FETCH);
     (002BH)  0000011  11  11  0101111  00000000  00  000  1  0  00010  00000000  0  0
```

```
XMAS VERS 2.0 PACED EMULATOR MICROCODE FOR INTEL 3000                                    ERRORS= 0   PAGE 23

RECORD               CPE     FI  FO   JUMP     KBUSS    PAGEF  CBUS  PAUSFF  STRB  LPCNT    EMIT    MULTF  SPRF
NUMBER             6543210  10  10  6543210  76543210   10     210    0       0   43210  76543210    0     0

 362  3BH!       LI!     NOP(A) JZR(FETCH);
      (003BH)  1100000  11  00  0101111  00000000    00     000    1       0   00010  00000000    0     0

 363  5BH!       SKP1!   NOP(A) JFL(POS1.NEG1);
      (005BH)  1100000  11  00  1000001  00000000    00     000    1       0   00010  00000000    0     0

 364  6BH!       SKP!    TZR(A) K80000 INH JCC(SKP1);
      (006BH)  1010000  11  00  0000101  10000000    00     001    1       0   00010  00000000    0     0

 365  7BH!       ANDD!   ACM(AC);
      (007BH)  0001011  11  00  0001000  00000000    00     000    1       0   00010  00000000    0     0

 366  8BH!               ANR(A) JZR(FETCH);
      (008BH)  1000000  11  00  0101111  11111111    00     000    1       0   00010  00000000    0     0

 367  0ABH!      SAME!   CMR(R8) JCR(DWN7);
      (00ABH)  1111000  11  00  0111101  00000000    00     000    1       0   00010  00000000    0     0

 368  0BBH!      CRY!    NOP(A) JFL(NOCRY2.CRY2);
      (00BBH)  1100000  11  00  1001101  00000000    00     000    1       0   00010  00000000    0     0

 369  0DBH!      CRY2!   NOP(A) FF1 STZ JZR(FETCH);
      (00DBH)  1100000  01  11  0101111  00000000    00     000    1       0   00010  00000000    0     0
```

```
370  0F8H!     ANEG2!  NOP(A)  JZR(FETCH);
     (00F8H)  1100000  11  00  0101111  00000000   00   000   1   0   00010  00000000   0   0

371  108H!     NSP!    NOP(A)  JZR(FETCH);
     (0108H)  1100000  11  00  0101111  00000000   00   000   1   0   00010  00000000   0   0

372  118H!     FS!     SDR(T)  FF1;
     (0118H)  0101100  11  11  0010010  11111111   00   000   1   0   00010  00000000   0   0

373  128H!             ILR(W);
     (0128H)  0000111  11  00  0010011  00000000   00   000   1   0   00010  00000000   0   0

374  138H!             SDR(A)  FF1;
     (0138H)  0100000  11  11  0010100  11111111   00   000   1   0   00010  00000000   0   0

375  148H!             ILR(T);
     (0148H)  0001100  11  00  0010101  00000000   00   000   1   0   00010  00000000   0   0
```

```
XMAS VERS 2.0 PACER EMULATOR MICROCODE FOR INTEL 3000                                          ERRORS= 0   PAGE 24

RECORD              CPE     FI  FO   JUMP     KRUSS    PAGEF  CBUS  PAUSFF  STRB  LPCNT    EMIT     MULTF  S2RF
NUMBER            6543210   10  10  6543210  76543210   10    210     0      0    43210  76543210    0      0

  376  15BH!             SDR(W) FF1 JZR(FETCH):
       (015BH)    0100111   11  11  0101111  11111111   00    000     1      0    00010  00000000    0      0

  377  16BH!      SKIP!  ILR(P) FF1 JZR(FETCH):
       (016BH)    0000011   11  11  0101111  00000000   00    000     1      0    00010  00000000    0      0

  378  17BH!      ODD!   NOP(A) JZR(FETCH):
       (017BH)    1100000   11  00  0101111  00000000   00    000     1      0    00010  00000000    0      0

  379  18BH!      CR!    NOP(A) JFL(OVEF2,NOVEF2):
       (018BH)    1100000   11  00  1001010  00000000   00    000     1      0    00010  00000000    0      0

  380  19BH!      OVEF1! ILR(P) FF1 JZR(FETCH):
       (019BH)    0000011   11  11  0101111  00000000   00    000     1      0    00010  00000000    0      0

  381  1ABH!      NOVEF2!                    NOP(A) JZR(FETCH):
       (01ABH)    1100000   11  00  0101111  00000000   00    000     1      0    00010  00000000    0      0

  382  1BBH!      CR22!  NOP(A) JFL(OVEF4,NOVEF4):
       (01BBH)    1100000   11  00  1001101  00000000   00    000     1      0    00010  00000000    0      0

  383  1CBH!      OVEF3! NOP(A) JZR(FETCH):
       (01CBH)    1100000   11  00  0101111  00000000   00    000     1      0    00010  00000000    0      0
```

```
384  1DBH!       NOVEE4!                     NOP(A) JZR(FETCH):
     (01DBH)   1100000  11  00  0101111   00000000   00    000    1    0    00010  00000000    0    0

385  1EBH!       NI2!     NOP(A) JZR(FETCH):
     (01EBH)   1100000  11  00  0101111   00000000   00    000    1    0    00010  00000000    0    0

386  3CH!        Iv!      LMI(R9) RRM:
     (003CH)   0011001  11  00  0000100   00000000   00    110    1    0    00010  00000000    0    0

387  4CH!                 ACM(AC):
     (004CH)   0001011  11  00  0000101   00000000   00    000    1    0    00010  00000000    0    0

388  5CH!                 ALR(X) JCR(CIX):
     (005CH)   0000001  11  00  0111101   11111111   00    000    1    0    00010  00000000    0    0

389  6CH!        S01!     NOP(A) JCC(S01):
     (006CH)   1100000  11  00  0011000   00000000   00    000    1    0    00010  00000000    0    0
```

```
RECORD               CPE     FI  FO    JUMP     KBUSS    PAGEF  CBUS  PAUSEF  STRB  LPCNT    EMIT    MULTF  SPRF
NUMBER             6543210   10  10  6543210  76543210    10    210     0      0    43210  76543210    0     0

 390  /*     REGISTER EIGHT IS SET TO STACK POINTER WHEN SIGNS OF MEMORY LOCAT
 391      AND ACCUMULATOR ARE DIFFERENT
 392
 393     REGISTER EIGHT IS SET TO THE DIFFERENCE OF THE MEMORY LOCATION AND THE
 394     ACCUMULATOR WHEN THE SIGNS OF MEMORY LOCATION AND ACCUMULATOR ARE DIFF
 395  ERENT   */
 396  7CH!       C!      ACM(AC);
      (007CH)  0001011  11  00  0001000  00000000    00    000     1      0    00010  00000000    0     0

 397  8CH!               SDR(R8) FF1;
      (008CH)  0101000  11  11  0001001  11111111    00    000     1      0    00010  00000000    0     0

 398  9CH!               SDR(T)  FF1 K80000;
      (009CH)  0101100  11  11  0001010  10000000    00    000     1      0    00010  00000000    0     0

 399  0ACH!              ILR(A) STC;
      (00ACH)  0000000  10  00  0001011  00000000    00    000     1      0    00010  00000000    0     0

 400  0BCH!              SDR(R5) FF1 K80000;
      (00BCH)  0100101  11  11  0001100  10000000    00    000     1      0    00010  00000000    0     0

 401  0CCH!              ILR(R5) FF0 STZ;
      (00CCH)  0000101  01  00  0001101  00000000    00    000     1      0    00010  00000000    0     0
```

```
402  0DCH!              XNR(T)  JCC(CX2)!
     (00DCH)  1111100  11  00  0010000  11111111   00   000   1   0   00010  00000000   0   0

403  0ECH!    CX3!     TZR(T)  INH!
     (00ECH)  1011100  11  00  0001111  11111111   00   001   1   0   00010  00000000   0   0

404  0FCH!             TZR(R5)  INH  JFL(NTSAME,SAME)!
     (00FCH)  1010101  11  00  1001010  11111111   00   001   1   0   00010  00000000   0   0

405  10CH!   CX2!  ILR(A)  FF0  JCC(CX3)!
     (010CH)  0000000  11  00  0001110  00000000   00   000   1   0   00010  00000000   0   0

406  11CH!      ICX!   ILR(R9)!
     (011CH)  0001001  11  00  0010010  00000000   00   000   1   0   00010  00000000   0   0

407  12CH!             ALR(X)!
     (012CH)  0000001  11  00  0010011  11111111   00   000   1   0   00010  00000000   0   0
```

```
XMAS VERS 2.0 PACER EMULATOR MICROCODE FOR INTEL 3000                                    ERRORS= 0   PAGE 26

RECORD                 CPE     FI  FO   JUMP     KBUSS    PAGEF  CBUS  PAUSFF  STRB  LPCNT    EMIT     MULTF  S2RF
NUMBER               6543210   10  10  6543210  76543210   10    210     0      0    43210  76543210    0      0

  408  13CH!                TZA(AC) K80000 INH!
       (013CH)      1011111   11  00  0010100  10000000   00    001     1      0    00010  00000000    0      0

  409  14CH!                NOP(A) JFL(NSKIP,SKIP)!
       (014CH)      1100000   11  00  1000110  00000000   00    000     1      0    00010  00000000    0      0

  410  16CH!        DWN3!   LTM(AC)!
       (016CH)      1011011   11  00  0010111  11111111   00    000     1      0    00010  00000000    0      0

  411  17CH!        SDR(A)  FF1 JZR(FETCH)!
       (017CH)      0100000   11  11  0101111  11111111   00    000     1      0    00010  00000000    0      0

  412  18CH!        SQ1!    TZR(A) K80000 INH JCF(NC,CR)!
       (018CH)      1010000   11  00  1010000  10000000   00    001     1      0    00010  00000000    0      0

  413  1ECH!        DI!     LMI(R9) RIN JCC(DWN3)!
       (01ECH)      0011001   11  00  0010110  00000000   00    100     1      0    00010  00000000    0      0

  414  3DH!         AA!     NOP(A) JZR(FETCH)!
       (003DH)      1100000   11  00  0101111  00000000   00    000     1      0    00010  00000000    0      0

  415  5DH!         CTX!    LMI(AC) RRM JPR(LA STA LX STX A1 SI M D AOM OR1 XOR1
  416                                      ANDD C SZ SNZ GG)!
       (005DH)      0011101   11  00  1100111  00000000   00    110     1      0    00010  00000000    0      0
```

```
417  6DH!      SN0!   NOP(A) JCC(SN01);
     (006DH)  1100000  11  00  0011001  00000000   00   000   1   0   00010  00000000   0   0

418  70H!       S2!    TZR(A)  INH;
     (0070H)  1010000  11  00  0001001  11111111   00   001   1   0   00010  00000000   0   0

419  9DH!              NOP(A) JFL(S1A,S2);
     (009DH)  1100000  11  00  1001001  00000000   00   000   1   0   00010  00000000   0   0

420  9AH!      S1A!   LMT(P) FF1 JZR(FETCH);
     (009AH)  0010011  11  11  0101111  00000000   00   000   1   0   00010  00000000   0   0

421  9BH!       S2!    NOP(A) JZR(FETCH);
     (009BH)  1100000  11  00  0101111  00000000   00   000   1   0   00010  00000000   0   0

422  8DH!      DWN5!  CMR(A) JZR(FETCH);
```

```
XMAS VERS 2.0 PACER EMULATOR MICROCODE FOR INTEL 3000                          ERRORS= 0   PAGE 27

RECORD               CPE     FI  FO   JUMP      KBUSS    PAGEF  CBUS  PAUSEF  STBB  LPCNT    EMIT    MULTF  SPRF
NUMBER             6543210   10  10  6543210  76543210    10    210     0      0    43210  76543210    0     0

         (009DH)   1110000   11  00  0101111  00000000    00    000     1      0    00010  00000000    0     0

 423   0ADH!       DWN7!  ALR(R8) FF1:
       (00ADH)     0001000   11  11  0001011  11111111    00    000     1      0    00010  00000000    0     0

 424   0BDH!       COXXX! TZR(R8) JFL(NOCRY.CRY):
       (00BDH)     1011000   11  00  1001011  11111111    00    000     1      0    00010  00000000    0     0

 425   11DH!       DCX!   ILR(R9):
       (011DH)     0001001   11  00  0010010  00000000    00    000     1      0    00010  00000000    0     0

 426   12DH!              CMA(AC):
       (012DH)     1111111   11  00  0010011  00000000    00    000     1      0    00010  00000000    0     0

 427   13DH!              ALR(X) FF1:
       (0130H)     0000001   11  11  0010100  11111111    00    000     1      0    00010  00000000    0     0

 428   14DH!              NOP(A) JFL(SP.NSP):
       (014DH)     1100000   11  00  1000000  00000000    00    000     1      0    00010  00000000    0     0

 429   19DH!       SNOI!  TZR(A) K80000 INH JCF(NC2 CR22):
       (019DH)     1010000   11  00  1010011  10000000    00    001     1      0    00010  00000000    0     0
```

```
430  100H!       DWN4!  LMI(R9) ROT JZR(FETCH);
     (0100H)    0011001  11  00  0101111  00000000  00  001  1  0  00010  00000000  0  0

431  1FDH!       D0!    ILR(A) JCC(DWN4);
     (01FDH)    0000000  11  00  0011101  00000000  00  000  1  0  00010  00000000  0  0

432  3FH!        COR!   NOP(A)  PAGE2 JCC(DWN8);
     (003FH)    1100000  11  00  0000101  00000000  01  000  1  0  00010  00000000  0  0

433  5FH!        DWN8!  NOP(A) JCC(SAF);
     (005FH)    1100000  11  00  0000110  00000000  00  000  1  0  00010  00000000  0  0

434  6FH!        SAF!   TZR(A) K00001 INH JCC(SAE1);
     (006FH)    1010000  11  00  0010110  00000001  00  001  1  0  00010  00000000  0  0

435  7FH!        SNZ!   TZR(A) INH;
     (007FH)    1010000  11  00  0001110  11111111  00  001  1  0  00010  00000000  0  0
```

```
RECORD              CPF     FI  FO   JUMP      KBUSS     PAGEF  CBUS  PAUSFF  STRB  LPCNT    EMIT      MULTF  S2PF
NUMBER            6543210   10  10  6543210   76543210    10     210     0      0    43210  76543210     0     0

 436  0FEH!               NOP(A) JFL(S3,S4);
      (00FEH)  1100000   11  00  1001110   00000000    00     000     1      0    00010  00000000     0     0

 437  0EAH!        S3!    NOP(A) JZR(FETCH);
      (00EAH)  1100000   11  00  0101111   00000000    00     000     1      0    00010  00000000     0     0

 438  0EBH!        S4!    LMI(P) FF1 JZR(FETCH);
      (00EBH)  0010011   11  11  0101111   00000000    00     000     1      0    00010  00000000     0     0

 439  11EH!      NOPP!    NOP(A) JZR(FETCH);
      (011EH)  1100000   11  00  0101111   00000000    00     000     1      0    00010  00000000     0     0

 440  16EH!      SAF1!    NOP(A) JFL(EVEN,030);
      (016EH)  1100000   11  00  1000111   00000000    00     000     1      0    00010  00000000     0     0

 441  1EEH!        HH!    NOP(A) JZR(FETCH);
      (01EEH)  1100000   11  00  0101111   00000000    00     000     1      0    00010  00000000     0     0

 442  0FH!      FETCH!    LMI(P) FF1 RRM;
      (000FH)  0010011   11  11  0000001   00000000    00     110     1      0    00010  00000000     0     0

 443  1FH!                LTM(AC) K01FFF;
      (001FH)  1011011   11  00  0000010   00111111    00     000     1      0    00010  00000000     0     0
```

```
444  2FH!              SDR(R9) FF1 JPX(DIR J L REG I JB UJT MSC XK JJ
445                                 CTRACK LL IX AA COMB IO);
     (002FH)  0101001  11  11  1111011  11111111   00    000    1     0    00010  00000000   0    0

446  3FH!      IO!     NOP(A) JCC(IO1);
     (003FH)  1100000  11  00  0010100  00000000   00    000    1     0    00010  00000000   0    0

447  6FH!      LAIBUS! LMI(R9) FF1 RRM;
     (006FH)  0011001  11  11  0000101  00000000   00    110    1     0    00010  00000000   0    0

448  5FH!               LDI(AC) FF1;
     (005FH)  0101111  11  11  0111000  11111111   00    000    1     0    00010  00000000   0    0

449  59H!               SDR(A) FF1 JZR(FETCH);
     (0059H)  0100000  11  11  0101111  11111111   00    000    1     0    00010  00000000   0    0

450  7FH!      GG!     NOP(A) JZR(FETCH);
```

```
XMAS VERS 2.0 PACER EMULATOR MICROCODE FOR INTEL 3000                                            ERRORS= 0   PAGE 29

RECORD                 CPE    FI  FO    JUMP      KBUSS     PAGEF   CBUS  PAUSFF   STPB  LPCNT    EMIT      MULTF  SPRF
NUMBER               6543210  10  10  6543210   76543210    10     210     0       0    43210  76543210     0      0

        (007FH)      1100000  11  00  0101111   00000000    00     000     1       0    00010  00000000     0      0

  451   11FH!        PZ!      NOP(A) SPO JZR(FETCH);
        (011FH)      1100000  11  00  0101111   00000000    00     000     0       0    00010  00000000     0      0

  452   14FH!        IO1!     NOP(A) JCC(JK);
        (014FH)      1100000  11  00  0011101   00000000    00     000     1       0    00010  00000000     0      0

  453   1DFH!        JK!      NOP(A) JPR(N1,N2,N3,N4,N5,TRAP,N7,RMI,SMI,EQ,N11,N12
  454                                    DI DO HH II);
        (01DFH)      1100000  11  00  1100110   00000000    00     000     1       0    00010  00000000     0      0

  455   1EFH!        II!      NOP(A) JZR(FETCH);
        (01EFH)      1100000  11  00  0101111   00000000    00     000     1       0    00010  00000000     0      0

  456   EOF

NO PROGRAM ERRORS
END OF PROGRAM
```

XMAS VERS 2.0 PACER EMULATOR MICROCODE FOR INTEL 3000 ERRORS= 0 PAGE 30

MICROPROGRAM MEMORY IMAGE

	0H	1H	2H	3H	4H	5H	6H	7H	8H	9H	AH	BH	CH	DH	EH	FH
000H	JCC 0010H 74 0		JZF 0012H 101 1	JZR 000FH 122 1												JCC 001FH 442 99
001H	JCC 0020H 75 1		JZR 000FH 102 1	JZR 000FH 123 1					JCC [illegible] 290 1		JZR 000FH 330 1	JZR 000FH 360 1				JCC 002FH 443 1
002H	JCC [illegible] 76 1		JZR 000FH 103 1	JZR 000FH 124 1			JCR 0027H 229 1	JCR 0028H 230 1	JCR 0029H 231 1	JZR 000FH 232 1	JZR 000FH 331 1	JZR 000FH 361 1				JPX 0030H 444 1
003H	[illegible] [illegible] 77 1	JZR 000FH 93 1	JCC 0042H 104 1	JCC 0103H 125 1	JCC 0044H 143 1	JCC 0045H 156 1	JCC 0046H 170 1	JPR 0060H 235 1	JCC [illegible] 289 1	JZR 000FH 312 1	JZR 000FH 333 1	JZR 000FH 362 1	JCC 004CH 396 1	JZR 000FH 414 1	JCC 005EH 432 1	JCC 014FH 445 1
004H	JCC [illegible] 78 1		JCC 0052H 105 1		JCC 0054H 144 1	JCC 0055H 157 1	JCC 0056H 171 1		JZR 0079H 291 1	JCR 004AH 313 1	JZR 000FH 334 1		JCC 005CH 397 1			
005H			JCR 0055H 106 1		JCR 0070H 145 1	JZR 000FH 158 1	JCR 0057H 172 1	JCR 0055H 237 1	[illegible]	JCC 0049H 314 1	JFL 0024H 335 1	JFL 001AH 363 1	JCR 005DH 398 1	JPR 0070H 415 1	JCC 006EH 433 1	JCR 005BH 448 1
006H	JCF 0022H [illegible] 1	JCC 0091H 94 1	JCF [illegible] 107 1	JCC 0083H 126 1	JCC 00F4H 147 1	JCC 0155H 159 1	JCC 0086H 173 1	JCC 0087H 238 1	[illegible] 1	JCC 0059H 315 1	JCC 005AH 336 1	JCC 005BH 364 1	JCC 019CH 399 1	JCC 019DH 417 1	JCC 016FH 434 2	JCC 005FH 447 1
007H	JCC [illegible] 81 4	JZR 000FH 95 4	JCC 0042H 109 4	JZR 000FH 127 4	JCC 0084H 149 4	JCC 0095H 160 4	JCC 0096H 175 4	JCC 00F7H 240 4	[illegible] 4	JCC 0089H 316 4	JCC 008AH 337 4	JCC 008BH 365 4	JCC 009CH 396 4	JCC 009DH 419 4	JCC 00EFH 435 4	JZR 000FH 450 4

MICROPROGRAM MEMORY IMAGE

	0H	1H	2H	3H	4H	5H	6H	7H	8H	9H	AH	BH	CH	DH	EH	FH
008H	JZR 000FH 82 1		JZR 000FH 109 1	JCF 00D2H 128 1	JZR 000FH 149 1	JCC 0095H 161 1	JCR 008FH 177 1	JCC 0097H 279 1	JCC 009AH 295 1	JZR 000FH 317 1	JCR 008DH 338 1	JZR 000FH 366 1	JCC 009CH 397 1	JZR 000FH 422 1	JCC 009EH 178 1	
009H	JCC 00A0H 83 1	JCF 0092H 96 1	JZF 00A2H 110 1	JZR 000FH 129 1		JZR 000FH 162 1	JCC 00A6H 181 1	JCC 00A7H 280 1	JCC 00AAH 296 1		JZR 000FH 420 1	JZR 000FH 421 1	JCC 00ACH 398 1	JFL 009AH 419 1	JCR 0096H 179 1	
00AH	JCC 00B0H 84 1		JZR 000FH 111 1	JZR 000FH 130 1	JCC 00B4H 185 1	JCR 00A4H 184 1	JCR 00A5H 183 1	JCC 00B7H 281 1	JCC 00BAH 297 1		JFL 00FAH 341 1	JCR 00ADH 367 1	JCC 00BCH 399 1	JCC 00BDH 423 1		
00BH	JCC 00C0H 85 1	JZF 00C2H 199 1	JCR 0081H 198 2	JCR 008FH 194 1	JCR 0085H 186 1	JCC 00C5H 187 1	JCC 00C6H 209 1	JCC 00C7H 282 1	JCC 00CAH 298 1		JZR 000FH 342 1	JFL 00DAH 368 1	JCC 00CCH 400 1	JFL 00BAH 424 1	JCR 00B2H 196 1	JCR 00BFH 195 1
00CH	JCC 00D0H 86 1	JCR 00CFH 207 1	JFL 00B2H 192 2	JCR 00C9H 201 1	JCR 00C2H 191 1	JCR 00C4H 189 1	JCC 00D6H 210 1	JCC 00D7H 283 1	JCC 00DAH 299 1	JFL 00CAH 202 1	JCR 00CEH 205 2	JCR 00CDH 203 1	JCC 00DCH 401 1	JCR 00CAH 204 1	JCR 00C1H 206 1	JZR 000FH 208 1
00DH	JCC 00E0H 87 1		JZR 000FH 112 1	JZR 000FH 131 1			JCC 00E6H 211 1	JCC 00E7H 284 1	JCC 00EAH 300 1		JZR 000FH 343 1	JZR 000FH 369 1	JCC 010CH 402 1			
00EH	JCC 00F0H 88 1		JZF 00F2H 113 1	JZR 000FH 132 1	JCF 00E2H 150 1		JCC 00F6H 212 1	JZR 000FH 285 1	JCC 00FAH 301 1		JZR 000FH 437 1	JZR 000FH 438 1	JCC 00FCH 403 1		JFL 00EAH 436 1	
00FH	JCC 0100H 89 1		JZR 000FH 114 1	JZR 000FH 133 1			JCC 0106H 213 1	JCC 0107H 241 1	JCC 010AH 302 1		JZR 000FH 344 1	JZR 000FH 370 1	JFL 00AAH 404 1			

XMAS VERS 2.0 PACER EMULATOR MICROCODE FOR INTEL 3000 ERRORS= 0 PAGE 32

MICROPROGRAM MEMORY IMAGE

	0H	1H	2H	3H	4H	5H	6H	7H	8H	9H	AH	BH	CH	DH	EH	FH
010H	JZR 000FH 90 1			JPR 0110H 134 1			JCC 0136H 214 1	JCC 0127H 242 1	JCC 0138H 363 1		JZR 000FH 345 1	JZR 000FH 371 1	JCC 00ECH 405 1			
011H	JZR 000FH 91 1	JZR 000FH 97 1	JCC 0122H 115 1	JCC 0123H 136 1	JCC 0124H 151 1	JCC 0125H 163 1	JCC 0126H 215 1	JZR 000FH 286 1	JCC 0128H 364 1	JCC 0129H 318 1	JCC 012AH 346 1	JCC 012BH 372 1	JCC 012CH 406 1	JCC 012DH 425 1	JZR 000FH 439 1	JZR 000FH 451 1
012H			JZR 000FH 116 1	JZR 000FH 137 1	JCC 0134H 152 1	JCC 0135H 164 1	JCC 0156H 216 1	JCC 0137H 243 1	JZR 000FH 365 1	JCC 0139H 319 1	JCC 013AH 347 1	JCC 013BH 373 1	JCC 013CH 407 1	JCC 013DH 426 1		
013H		JZR 000FH 98 1	JCR 0131H 117 1	JCR 0134H 138 1	JFL 0132H 153 2	JCC 0145H 165 1	JCC 0146H 217 1	JCC 0147H 244 1	JCC 0148H 366 1	JCC 0149H 320 1	JCC 014AH 348 1	JCC 014BH 374 1	JCC 014CH 408 1	JCC 014DH 427 1		
014H			JZR 000FH 118 1	JCR 0145H 139 1		JFL 0142H 166 2	JCC 0166H 218 1	JCC 0157H 245 1	JCC 0158H 367 1	JCC 0159H 321 1	JCC 015AH 349 1	JCC 015BH 375 1	JFL 016AH 409 1	JFL 010AH 428 1		JCC 010FH 452 1
015H		JZR 000FH 99 1	JCR 0151H 220 1	JCR 0156H 221 1		JCF 0162H 167 1	JFL 0152H 219 2	JCC 0167H 246 1	JCC 0168H 368 1	JZR 000FH 322 1	JZR 000FH 350 1	JZR 000FH 376 1				
016H			JZF 0172H 119 1	JZR 000FH 140 1			JCC 0176H 222 1	JCC 0177H 247 1	JZR 000FH 369 1		JZR 000FH 351 1	JZR 000FH 377 1	JCC 017CH 410 1		JFL 017AH 440 1	
017H			JZR 000FH 120 1	JZR 000FH 141 1			JCC 0186H 223 1	JCC 0187H 248 1			JZR 000FH 352 1	JZR 000FH 378 1	JZR 000FH 411 1			

XMAS VERS 2.0 PACER EMULATOR MICROCODE FOR INTEL 3000 FRRORS= 0 PAGE 33

MICROPROGRAM MEMORY IMAGE

	0H	1H	2H	3H	4H	5H	6H	7H	8H	9H	AH	BH	CH	DH	EH	FH
018H	JFL 0192H 255 1	JCR 0180H 254 1	JCR 0181H 253 2	JCR 0188H 275 1	JCR 0182H 251 1	JCR 0184H 250 1	JCC 0196H 224 1	JCR 0185H 249 1	JCC 0198H 276 1	JCC 0199H 323 1	JFL 019AH 353 1	JFL 01AAH 379 1	JCF 018AH 412 1			
019H	JCC 01A0H 259 1	JCR 0190H 258 1	JCR 0191H 257 1	JCR 0194H 264 1	JCR 0195H 265 1	JCR 0197H 266 1	JCC 01A6H 225 1	JCC 01A7H 267 1	JCC 01A8H 277 1	JCC 01A9H 324 1	JZR 000FH 354 1	JZR 000FH 380 1		JCF 018AH 429 1		
01AH	JCC 0180H 260 1						JCC 0186H 226 1	JCC 0187H 268 1	JZR 000FH 278 1	JCC 0189H 325 1	JZR 000FH 355 1	JZR 000FH 381 1				
01BH	JCC 01C0H 261 1	JCC 01C1H 270 1					JCC 01C6H 227 1	JCR 01B1H 269 1		JCC 01C9H 326 1	JFL 01CAH 356 1	JFL 01DAH 382 1				
01CH	JCR 01C1H 262 1	JCR 01C7H 272 2					JCC 0026H 228 1	JFL 0182H 273 1		JZR 000FH 327 1	JZR 000FH 357 1	JZR 000FH 383 1				
01DH					JCR 01D9H 154 1	JCR 01D4H 168 1	JCR 01D5H 234 0	JZR 000FH 287 1	JCR 01D7H 310 1	JCR 01D8H 328 1	JZR 000FH 358 1	JZR 000FH 384 1		JZR 000FH 430 1		JPR 01F0H 453 1
01EH	JZR 000FH 92 1	JZR 000FH 100 1	JZR 000FH 121 1	JZR 000FH 142 1	JZR 000FH 155 1	JZR 000FH 169 1	JZR 000FH 233 1	JZR 000FH 288 1	JZR 000FH 311 1	JCC 0189H 329 1	JZR 000FH 359 1	JZR 000FH 385 1	JCC 015CH 413 1	JCC 01DDH 431 1	JZR 000FH 441 1	JZR 000FH 455 1
01FH																

XMAS VERS 2.0 PACER EMULATOR MICROCODE FOR INTEL 3000 ERRORS= 0 PAGE 34

CROSS REFERENCE DIRECTORY

LABEL	REFERENCES
A1	77,145,(148),291,415
AA	(414),445
AGN	219,(221)
AGN2	(138),153
AGN3	(139),166
ALS	134,(163
ANDD	78,146,202,(365),416
ANEG2	341,(370
ADA	(97),134
ADM	77,145,201,(294),415
APOS2	341,(344
ARS	134,(151)
BACK	139,(166)
C	78,146,202,(396),416
CAD	(115),134
CIX	388,(415
CL	106,(158,,237
CLI	172,(237)
CLR1	(91),134
COR	(432),445
COXXX	(424)
CR	(379),412
CR22	(382),429
CRY	(368),424
CRY2	368,(369
CTRACK	(333),445
CX2	402,(405,
CX3	(403),405
D	77,145,(240),291,415
DCX	135,(425
DI	(413),454
DIR	(77),444
DIV	(253),273
DNI	(323),320
DO	(431),454
DONE	273,(275)
DWN1	(154),160
DWN2	(168),236
DWN3	(410),417
DWN4	(430),431
DWN5	338,(422,
DWN7	367,(423)
DWN9	432,(433)
F2	135,(346)
FO	(329),453
FO3	107,(122
FO5	(132),155
FO6	(143),167
FOL	90,(124)
FOL2	96,(129)
FOL4	128,(131)
FS	135,(372)

XMAS VERS 2.0 PACER EMULATOR MICROCODE FOR INTEL 3000 ERRORS= 0 PAGE 35

CROSS REFERENCE DIRECTORY

```
LABEL    REFERENCES

EVEN     (352),440
EX       135,(318)
FETCH    82,90,91,92,93,95,97,98,99,100,102,103,109,111,112,114,116,118,120,121,122,123,124,127,129,130,131,132,133,137,140,141,
         142,149,155,158,162,169,208,232,233,278,285,286,287,288,294,306,309,311,312,317,322,327,330,331,333,334,342,343,344,345,
         350,351,352,354,355,357,358,359,360,361,362,366,369,370,371,376,377,378,380,381,383,384,385,411,414,420,421,422,430,437,
         438,439,441,(442),449,450,451,455
GG       78,146,292,416,(450)
GT       110,(130
GT5      113,(133)
GT6      119,(141)
HH       (441),454
I        (143),444
ICX      135,(406
II       454,(455)
INIT     (74)
INIT1    76,(79)
INIT2    79,(83)
IO       445,(446)
IO1      446,(452)
IX       (386),445
J        (93),444
JI       (156),444
JJ       (312),444
JK       452,(453)
L        (104),444
LA       77,(81),145,291,415
LAIBUS   236,(447
LI       (170),444
LL       (362),445
LRS      134,(215,
LT       101,(102)
LX       77,(108),145,291,415
M        77,145,(175),291,415
MB1      192,(194
MB2      192,196,(198)
MEX      199,(201)
MLP      (192),199
MSC      (235),444
N1       (92),453
N11      (359),453
N12      (385),453
N2       (100),453
N3       (121),453
N4       (142),453
N5       (155),453
N7       (233),453
NC       (353),412
NC2      (356),420
NEG      202,(203,
NEG1     (360),363
NEGG     335,(341
NOCRY    (342),424
```

```
XMAS VERS 2.0 PACER EMULATOR MICROCODE FOR INTEL 3000                                ERRORS= 0   PAGE 36

CROSS REFERENCE DIRECTORY

LABEL      REFERENCES

NOCRY2     (343),368
NOPP       135,(439
NOVEF1     353,(354)
NOVEF2     379,(381)
NOVEF3     356,(357)
NOVEF4     382,(384)
NSKIP      (351),402
NSP        (371),428
NTEQ3      (101),107
NTEQ5      (113),156
NTEQ6      (119),167
NTEQL      90,(103)
NTEQL2     96,(112)
NTEQL4     (112),128
NTGT       110,(111)
NTGT5      113,(114
NTGT6      119,(120)
NTLT       101,(123)
NTSAME     (341),406
OOD        (378),44
OR1        77,145,201,(316),415
OUT        219,(220
OUT2       (117),153
OUT3       (118),166
OV1        194,(195
OV2        203,(204)
OV3        205,(206)
OVEF1      353,(380)
OVEF2      (355),372
OVEF3      356,(383
OVEF4      (358),382
OVF2       262,(272
POPX       235,(293)
POPX1      293,(295)
POPX2      303,(316)
POS        202,(205)
POS1       (330),362
POSS       (331),335
PUSA       (173),235
PUSA2      214,(217
PUSA3      218,(222)
PUSA4      154,(328)
PUSA5      (312),328
PUSA6      (297),310
PUSAC      173,(209
PUSX       235,(238)
PUSX1      238,(279)
R7         135,(451)
REG        (125),444
REGC       125,(134
RMI        (288),453
RTN        236,(315)
```

```
XMAS VERS 2.0 PACER EMULATOR MICROCODE FOR INTEL 3000                                  ERRORS= 0   PAGE 37

CROSS REFERENCE DIRECTORY

LABEL     REFERENCES

RTN1      (314),315
RTN2      (313),314
RTN3      313,(334
S1        77,145,(160),291,415
S1A       419,(420
S2        419,(421)
S3        436,(437)
S4        436,(438)
SAE       236,433,(434)
SAE1      434,(440)
SAME      (367),404
SE        (80),235
SG        (94),235
SGC       94,(96)
SGE       (147),235
SGEC      147,(150
SKIP      (377),409
SKN       236,(336)
SKN1      (335),334
SKP       236,(364)
SKP1      (363),364
SL        (107),235
SLE       (159),235
SLEC      159,(167
SMI       (311),453
SNE       (126),235
SNEC      126,(128
SNO       236,(417)
SNO1      417,(429)
SNZ       78,146,292,416,(435)
SO        236,(389)
SO1       389,(412)
SONE      255,(264)
SP        (345),428
SSN       135,(304)
SSP       134,(286)
STA       77,(95),145,291,415
STX       77,(127),145,291,415
SZ        78,146,292,416,(418)
SZERO     255,(257
TCA       134,(136
TRAP      (169),453
WW        (99),220
XK        (289),446
XOR1      77,145,291,(337),415
XX        138,(153
XXX       201,(202)
YY        (98),117
ZZ        216,(219),221
```

DISTRIBUTION

	No. of Copies
Defense Documentation Center Cameron Station Alexandria, VA 22314	12
Deputy Chief of Research and Development Department of the Army Washington, D. C. 20310	1
Office, Assistant Chief of Staff for Force Development Department of the Army ATTN: FORSD, MG W. A. Burke	1
FORSD-FS, COL J. N. Hale Washington, D. C. 20310	1
Commander US Army Combat Developments Command Field Artillery Agency ATTN: COCFA-WC, F. Shelton Fort Sill, Oklahoma 73503	1
Commander US Army Weapons Command ATTN: AMSWE-RD, COL C. Alter	1
J. Brinkman	1
AMSWE-RDP, Dr. M. Amoruso	1
AMSWE-RDF, G. Schlenker	1
AMCPM-CAWS, H. Noble	1
SWEER-R-ST, H. Garver Rock Island, Illinois 61201	1
Director Aberdeen Research and Development Center ATTN: AMXRD-BAD, Dr. R. Eichelberger Aberdeen Proving Ground, Maryland 21005	2
Commander Picatinny Arsenal ATTN: SMUPA-DWO, J. Brooks Dover, New Jersey 07801	1
Commander US Army Material Command ATTN: DRCRD-MT-MT 501 Eisenhower Avenue Alexandria, Virginia 22304	5
Director Night Vision Laboratory (US AECOM) ATTN: Dr. John Dehue DRESL-NV-VI Dr. George Jones Fort Belvoir, Virginia 22060	2

	No. of Copies
US Army Materiel Systems Analysis Activity ATTN: DRXSY-MP Aberdeen Proving Ground, Maryland 21005	1
University of California Dept. of Electrical Engineering ATTN: Dr. Jack Sklansky Irvine, California 92664	2
US Army Research Office ATTN: Information Processing Office Box CM, Duke Station Durham, NC 27706	1
Defense Advanced Research Project Agency ATTN: Maj. Karam	1
Dr. Tegnelia 1400 Wilson Boulevard Arlington, Virginia 22209	1
Martin Marietta Corp. ATTN: A. C. Layton P. O. Box 5837 MP 251 Orlando, Florida 32805	1
Texas Instruments, Inc. ATTN: L. J. Freeman	1
Dr. Clay Shultz P. O. Box 6015 Dallas, Texas 75222	1
Dr. Dave Casasent Carnegie-Mellon University Pittsburgh, PA 15213	1
DRSMI-LP, Mr. Voigt	1
-W	1
DRDMI-X Commander	1
-T, Dr. Kobler	1
-T, Mr. James J. Fagan	1
-TE, Mr. Lindberg	1
-TE, Mr. Pittman	1
-TEO, Dr. Minor	25
Mr. Sitton	1
Mr. Widenhofer	1
Mr. Wylie	1
Mr. Farmer	1
Dr. Kulas	1
-C, Maj. W. Ward	1
Maj. W. Adams	1
-TG, Mr. J. Huff	1
-HF, Mr. N. Mangns	1
-TD, Dr. McCorkle	1
-TI, (Record Set)	1
(Reference Set)	1